DOCUMENT RESUME

ED 246 088 TM 840 360

AUTHOR Bowers, John J.
TITLE Database Management for Item Banking and Test Development: An Application of dBase II for the Microcomputer.
PUB DATE Apr 84
NOTE 129p.; Paper presented at the Annual Meeting of the American Educational Research Association (68th, New Orleans, LA, April 23-27, 1984).
PUB TYPE Speeches/Conference Papers (150) -- Reports - Descriptive (141)

EDRS PRICE MF01/PC06 Plus Postage.
DESCRIPTORS *Computer Software; Databases; *Item Banks; *Microcomputers; *Test Construction; Test Items
IDENTIFIERS *Database Management Systems; *dBase II; IBM Personal Computer

ABSTRACT

The background and results of an effort to use dBASE II, a microcomputer database management package, to establish, maintain, and update an item bank useful in a complex test development process are presented. The paper explores some of the perspectives and considerations in designing such a database which make the test development process easier, while fully utilizing the software's capabilities. It discusses principles of database design, features and advantages of a relational database schema, the structure of database files useful in test development, and a general test development process. Sample programs and output from a model system, dubbed ITEMBANK, are presented. The paper consists of a brief summary of the development effort and several extensively annotated figures. Figures discuss the nature of database design and the particular advantages of a relational database structure; describe a "dummy" database representing a microcosm of a test development effort for various forms of a fictional test used in developing the ITEMBANK programs; and discuss the four-stage test development process the system was designed to serve. The final figures concern the construction of the ITEMBANK programs and present sample printouts. Appendices list the database and programs for the draft test stage of the ITEMBANK system. (Author/DWH)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED246088

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

☒ This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

Database Management for Item Banking and Test Development: An Application of dBASE II for the Microcomputer

John J. Bowers, Pn.D.
The American College Testing Program

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

J. J. Bowers

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

Presented at the Annual Meeting of the
American Educational Research Association
New Orleans, Louisiana
April 23-27, 1984

TM 840 360

Database Management for Item Banking and Test Development: An Application of dBASE II for the Microcomputer

John J. Bowers, Ph.D.
The American College Testing Program

ABSTRACT

This paper presents the background and results of an effort to use dBASE II, a popular microcomputer database management package, to establish, maintain, and update an item bank useful in a complex test development process. It explores some of the perspectives and considerations in designing such a database so as to make the test development process easier while using the software's capabilities to the fullest. It discusses principles of database design, the features and advantages of a relational database schema, the structure of database files useful in test development, and a general test development process, and presents sample programs and output from a model system, dubbed ITEMBANK.

The paper consists of a brief summary of the development effort, followed by twenty extensively annotated figures. Figures 1 to 7 discuss the nature of database design and the particular advantages of a relational database structure. Figures 8 to 11 describe a "dummy" database representing a microcosm of a test development effort for a number of forms of a fictional test that was used in developing the ITEMBANK programs. Figures 12 to 14 discuss the four-stage test development process that the ITEMBANK system was designed to serve. And Figures 15 to 20 concern the construction of the ITEMBANK programs and present sample printouts from them. Appendix A lists the entire dummy database and Appendix B presents all programs for the draft test stage of the ITEMBANK system, written in dBASE II for the IBM PC.

Database Management for Item Banking and Test Development: An Application of dBASE II for the Microcomputer

John J. Bowers, Ph.D.
ACT, Inc.

Summary

Overview

This paper presents the background and results of an effort to use a popular microcomputer database management package to establish, maintain, and update an item bank useful in a complex test development process. It explores some of the perspectives and considerations in designing such a database so as to make the test development process easier while using the software's capabilities to the fullest. It discusses principles of database design, the features and advantages of a relational database schema, the structure of database files useful in test development, and a general test development process, and presents sample programs and output from a model system.

Perspectives

Imagine the following situation, much or all of which is not uncommon to a number of test development situations faced by researchers and evaluators: You have the task of developing a test which will be given once or twice a year for a particular client (e.g., a school district, corporation, or professional organization). The test is composed of, say, 150 items that are drawn from a larger item bank of 1,000 items. Because a different group takes the test each time it is administered, and because you need to equate scores from one test to another to ensure that the interpretation of a given score remains constant, you need to repeat a number of items from one test form (or forms) to another. A table of specifications determines how many items are assigned to each content classification and skill area on the test. In addition to meeting these content specifications for each form of the test, you must also select items so that each form (and, to some extent, subsections within each form) meet certain performance requirements with respect to difficulty and discrimination. Further, content experts may review and drop outdated or poorly performing items from the item bank, revise others, and add new items, so that the item bank is constantly changing. Finally, item performance data from current test forms must be added and the item bank updated with the completion of each test development cycle.

Current statistical software for both microcomputers and mainframes are not designed to handle such complex data management tasks easily. Database management packages (e.g., INFO) present problems of availability and, when available, of test security since they are usually used on large, publicly accessible computer systems. Their use may also require sophistication in the use of computer systems on the part of clerical and professional test development staff.

Such problems can be avoided when using a database management package on a microcomputer. The hardware, software, and item bank data are contained entirely within the test development organization, secure yet easily accessible and usable. Often, such packages are more easily used by persons with

little or no experience in computer use and programming. With these considerations in mind, this writer designed and developed an item banking system using the data manipulation and programming facilities of dBASE II, a popular and widely used database management system for microcomputers.

Several considerations influenced the choice of dBASE II and the design of the system. First, it had to be possible to integrate the software (and the programs written in it) with the total test development process. It could not be simply a computerized version of a card file. It had to be able to print reports, produce new data files or copies of files, update data files, print reports, and so on. Second, the system had to be "user-friendly." Many of the personnel who would use the system would not have had experience with computers or programs. This meant that the system had to have prepared programs accessed through menus and prompts that could allow a task to be accomplished with the push of a few keys. Such menus and prompts would also allow more uniform input, processing, and output of data, with fewer errors due to individual differences in system users. Third, the system had to be flexible, i.e., to be able to access and organize different files. This proved to be one of dBASE II's most important advantages: It can handle several files, joining data from them as needed and using one file to organize data in another. Other files contain programs for processing data, and formats for customizing and controlling data entry and reports.

Much of dBASE II's power comes from the fact that it is a relational database system. It stores data in a series of flat two-dimensional tables or files (called relations in database terminology). Data from different files can be joined so long as they possess one or more key fields in common. Other schema, such as the single flat table file (e.g., the matrix files used by statistical packages) or the hierarchical database (typical of census data) have certain disadvantages. Figures in the paper explain and compare these three systems in more detail.

Thus, an item bank based on a relational database system can have separate files: one with master data for items, consisting of one record for each item in the pool; another with one record for each usage of an item; and still others maintaining other data useful in the test development process. Such an arrangement packs a considerable amount of data into a relatively small space. The software permits the user to join the master item data with all the usage data or with only the data for selected uses, or for selected forms, and only for those variables needed for a particular purpose. Such joining can be done on a temporary basis during the course of a program, but the results can also be saved as an entirely new database file. In addition to its file manipulation capabilities, dBASE II is a programming language (as contrasted to a set of commands like SPSS) so that programs can be run with a simple command like "DO program" where "program" is the name of a large, complex program that will process certain files, print a report, and then branch to still other programs.

Methods and Techniques

The practical problems and possibilities in developing such a database management system were explored by (1) constructing a "dummy" database representing a microcosm of a test development effort for a number of forms of a fictional test, and (2) developing a set of programs, dubbed ITEMBANK, that would illus-

trate the use of dBASE II in a test development process relating to the fictional test. Considerable care was taken that both the dummy database and the test development process reflect real-world test development situations.

Two data files were established in the dummy database. In each file, an item was identified by a unique item identifier related to an item's physical position on a magnetic disk in a word processing system. Thus, the item identifier acts as a stand-in for the item itself. The master file (called POOL) contained data that is relatively stable over a number of uses of an item. In addition to the item identifier, the file contained such variables as the item's classification within the table of test specifications, its format (multiple-choice, option, etc.), its key, its length in lines of text, its current status (active or inactive), and other fields, most notably those which indicate its status in the current test development cycle. The usage data file (called HIST) contained one record for each use of an item on a test form. In addition to the unique item identifier, each record contained such variables as the form on which the item was used, its number on that form, its key on that form, its function (e.g., as an equator item, a new item, etc.), and its difficulty and discrimination. There were 124 records in the POOL file and 280 records in the HIST file. These represented the use and re-use of 124 items across 7 forms of a (fictional) forty-item test. Test characteristics (e.g., content breakdown, performance, percent of items allocated to certain functions) were designed to reflect real-world test development requirements.

To guide the design of task menus, as well as the input, processing, and outputs of programs, the test development process was conceptualized as a series of steps, selected steps being assisted by the dBASE II item-banking system. Because there were many steps, and different requirements with respect to data processing from one step to another, the test development process was organized into four stages: the Draft Test Stage, the Working Test Stage, the Final Test Stage, and the Item Analysis and Updating Stage. Each stage had associated with it a menu of tasks, called steps, programs accessed by the menu in order to process the item bank data in specified ways, and reports and printouts. The Draft Test Stage, for example, presents a menu that permits the user to obtain printouts of the complete item history data for all items eligible for the next test form., to create a new file consisting of all items tentatively selected for the new test form, to print important data regarding those items for further use by test developers, to replace items in the draft test file so as to achieve certain test characteristics, to create a list of eligible alternate items that may be used by content reviewers to further modify the test, and so on.

The programs and database were implemented on an IBM PC with color monitor. They make full use of the capabilities of dBASE II as used on the PC, including the extensive use of color in presenting numerous data entry screens. The paper presents a complete listing of the contents and structure of the dummy database, the source programs for the Draft Test Stage, and sample printouts from those programs. Additional information about the use of the database and programs is available from the author: John J. Bowers, Ph.D., The American College Testing Program, Health Programs Department, Area 29, PO Box 168, Iowa City, Iowa 52243, phone (319) 337-1063.

LIST OF FIGURES

Figures		Page
1.	Definition of a Database	5
2.	Illustrative Single Flat File Database Structures	6
3.	Illustrative Hierarchical File Database Structure	8
4.	Normalizing the Illustrative Database Structure	10
5.	Structuring Relational Database Files Through Normalization	11
6.	Some Advantages of Normalized Data Structures and Relational Databases	13
7.	Questions That Might Be Answered Using a Relational Item Bank Database	14
8.	Description of the Dummy Database Constructed to Test ITEMBANK Programs	15
9.	POOL File Structure	18
10.	HIST File Structure	22
11.	Other Database Files Useful in an Item Bank	26
12.	Considerations in the Development of the ITEMBANK System Using dBASE II	29
13.	The Four Stages of Test Development in the ITEMBANK System: An Overview	30
14.	The Test Development Process in the ITEMBANK System	31
15.	ITEMBANK Main Menu Program Branching	35
16.	Draft Stage Menu Program Branching	36
17.	Sample Printout of a Page from the ITEMBANK Test Drafting Kit	38
18.	Sample Printouts of a Test Summary Sheet from the ITEMBANK Draft Test Stage	40
19.	Sample Printout of an Alternate Items List from the ITEMBANK Draft Test Stage	45
20.	Considerations in Writing the ITEMBANK Programs	48

APPENDICES

A.	Listing of the ITEMBANK Dummy Database	A-1
B.	The ITEMBANK System Draft Stage Programs	B-1

Figure 1

Definition of a Database*

Collection of interrelated data

Multiple users

Stored independent of user programs

Common controlled approach for:

- Adding data
- Deleting data
- Updating data

Controlled by DBMS

*from Schussel (no date).

Figure 2

Illustrative Single Flat File Database Structures

Assume a very simple database with only 10 items that have been used to construct 4 forms of a 4-item test. In each item record, each cell represents a series of fields giving data of interest about the item. The field labeled "Master Data" contains basic data about the item, that changes little, if at all, from one use of the item to the next (e.g., its content classification, format, etc.). Each of the remaining cells represents item usage data, i.e., data that varies from use to use (e.g., test form number, difficulty, discrimination, etc.). Two types of flat files are shown, the first organized by test form number, the second by use. Dashes (--) indicate the cell is

Item ID	Master Data	Form 1	Form 2	Form 3	Form 4
A	data	--	--	Use 1	--
B	data	--	Use 1	--	--
C	data	Use 1	Use 2	--	Use 3
D	data	--	--	Use 1	Use 2
E	data	Use 1	--	Use 2	--
F	data	--	Use 1	--	--
G	data	--	Use 1	Use 2	--
H	data	Use 1	--	--	Use 2
I	data	Use 1	--	--	--
J	data	--	--	--	Use 1

Item ID	Master Data	Use 1	Use 2	Use 3	Use 4
A	data	Form 3	--	--	--
B	data	Form 2	--	--	--
C	data	Form 1	Form 2	Form 4	--
D	data	Form 3	Form 4	--	--
E	data	Form 1	Form 3	--	--
F	data	Form 2	--	--	--
G	data	Form 2	Form 3	--	--
H	data	Form 1	Form 4	--	--
I	data	Form 1	--	--	--
J	data	Form 4	--	--	--

Easy to understand for most people.

Top form is relative easy to maintain. Bottom form is very difficult to maintain and update.

Both forms waste a lot of space (although bottom form sometimes gives the illusion of saving space).

Both forms are difficult to expand. For example, if the Master Data cell comprised 11 fields, and each form/use cell comprised 7 fields, only three uses or forms would use up the dBASE II limit of 32 fields. Other data of interest (e.g., the item writer, the item reference, comments) could not easily be

maintained in these forms or related to item and test data.

Some questions are more easily answered with one form than the other. The top style is good for reconstructing test forms, the bottom style for reconstructing item history. Both forms would make it difficult to answer questions such as "What items have a difficulty between .50 and .90 and discrimination greater than .20?"

Figure 3

Illustrative Hierarchical File Database Structure

This illustration uses the same data as the previous illustration: a pool of 10 items used in 4 test forms of 4 items each. Again, each cell (master data and form number) represents a set of fields. There are two record types: 1 = master data record, 2 = usage data record.

Item ID	Record Type	Record Content
A	1	Master data
A	2	Form 2
B	1	Master data
B	2	Form 2
C	1	Master data
C	2	Form 1
C	2	Form 2
C	2	Form 4
D	1	Master data
D	2	Form 3
D	2	Form 4
E	1	Master data
E	2	Form 1
E	2	Form 3
F	1	Master data
F	2	Form 2
G	1	Master data
G	2	Form 2
G	2	Form 3
H	1	Master data
H	2	Form 1
H	2	Form 4
I	1	Master data
I	2	Form 1
J	1	Master data
J	2	Form 4

More economical of space than the flat file organization on the previous page, but needs an extra field (record type) for each record.

Software not that common, especially for microcomputers. Does not usually permit interactive queries. Complicated, time-consuming, expensive to use.

Not very flexible. Difficult to expand and update files. Better for static files that do not change much. Like flat files, other data of interest (e.g., item writer, item reference, comments, etc.) are difficult to maintain in this form.

Best searched from top to bottom. Best searched for particular records (many to one). More difficult to do "many to many" type searches.

"...one of the drawbacks of the hierarchical model [is] unnecessary complexity. Specifically, the user is forced to devote time and effort to solving

problems which are introduced by the model and are not intrinsic to the questions being asked." (Martin, 1976)

Figure 4

Normalizing the Illustrative Database

This illustrates the same 10-item database, now normalized into two files and ready to be used as a relational database.

File Name: POOL
Primary Key: Item ID

Item ID	Master Data
A	data
B	data
C	data
D	data
E	data
F	data
G	data
H	data
I	data
J	data

File Name: HIST
Primary Keys: Item ID, Form #

Item ID	Form #	Usage Data
A	Form 3	data
B	Form 2	data
C	Form 1	data
C	Form 2	data
C	Form 4	data
D	Form 3	data
D	Form 4	data
E	Form 1	data
E	Form 3	data
F	Form 2	data
G	Form 2	data
G	Form 3	data
H	Form 1	data
H	Form 4	data
I	Form 1	data
J	Form 4	data

The entire database now exists as two flat files. The POOL file contains only master data on each item, one record per item keyed on a unique ID number. The HIST file contains all usage data, one record for each use of each item, keyed on the item's unique ID number and the number of the test form on which it was used. Within each file, no two records have the exact same key.

Files are easier to update. Simply append new data to the "bottom" of the existing files. Records may be added in any order. The physical organization is independent of the logical organization, i.e., the way in which the data is presented to the user.

It is now possible, using a relational DBMS such as dBASE II, to relate data in one file to data in another file, making it possible to have many such files with data of interest (see Figure 11). Data from one file can be joined to another by use of the primary keys.

Figure 5

Structuring Relational Database Files
Through Normalization

Questions to be answered in structuring database files:

1. How many flat table files do you need?

2. What is to be the logical structure of each file, i.e., what data items (records and fields) will be in each file?

Because the data management software can keep the user's view of the data (i.e., the logical structure) entirely separate from the physical structure of the data, one consideration is of overriding importance in the logical data description: the convenience of the majority of application programmers and users.

Thus, the logical structure should:

1. be easily understood by users with no training in programming;

2. make it possible to add records to the database without changing the existing logical structure, and hence changing the application programs; and

3. permit maximum flexibility in formulating unanticipated or spontaneous inquiries.

The most natural way to do this is with two-dimensional tables since they are easier to remember and visualize.

Normalization

Normalization is a step-by-step process for replacing complicated data relationships with relationships in flat two-dimensional tabular form (without, of course, losing information about relationship between data).

Such tables have the following properties.

1. Each entry in a table represents one data item; there are no repeating groups.

2. They are column-homogenous; i.e., in any column all items are of the same kind.

3. Each column is assigned a distinct name.

4. All rows are distinct; duplicate rows are not allowed.

5. Both the rows and the columns can be viewed in any sequence at any time without affecting either the information content or the semantics of any function using the table.

Such tables are called relations, thus the name relational database. (More jargon: A field is an attribute. A record is a tuple. An attribute for a tuple is a data item. All data items for a given attribute constitute a domain.)

Remember, ALL FILES CAN BE REPRESENTED AS FLAT FILES.

The eventual objective of normalization is a third normal form. This involves selecting attributes in such a way that the records will be stable as the database grows, changes, and is used for new applications.

An informal definition of a third normal form is that a relation (flat file) is third normal form when:

1. nonkey attributes are mutually independent;

2. nonkey attributes are fully dependent on the primary key(s); and

3. there are no repeating groups, only atomic (i.e., nondecomposable) values.

Thus, designing a relational database involves creating user views of how the data is to be displayed, decomposing these into third normal form, and defining these relations to the system (in dBASE II, the CREATE command).

References: Martin (1976), an excellent non-technical introduction to database theory, was the source for most of the information here. Other references that were used, and that can provide the reader with more detailed discussions of the above, include Martin (1975), Date (1975), Kent (1983), and Codd (1982).

Figure 6

Some Advantages of Normalized Data Structures
and Relational Databases

1. Clarity and ease of use. Two-dimensional tables are the easiest way to represent most data to users not trained in data processing. With the right DBMS, database access is much more user-friendly.

2. Flexibility. Tables can be "cut and pasted" together so that different logical files wanted by different users can be given to them in the form they want.

3. Security. Always a concern in test development, it can be more easily implemented for files with sensitive attributes such as the master item key. Security passwords or other requirements can be place on selected files or the sensitive attributes can be placed in separate files and retrieved only when security conditions are met.

4. Relatability. The maximum flexibility is possible in relating attributes from different files.

5. Ease of implementation. The physical storage of flat files can be less complex than that of other file types.

6. Data independence. There will be need for most databases to grow by adding new attributes and relations. The data will be used in new ways. If the database is in a normalized form with data independence in the software, the database can grow without, in most cases, forcing the rewriting of application programs. This can lower the cost of data maintenance and software development. This the objective of having a clear separation between the logical and physical aspects of the database.

*Abstracted from Martin (1976). Martin lists several others not mentioned here that deal with the theory and mathematics of relational databases.

Figure 7

Questions That Might Be Answered
Using A Relational Item Bank Database

Relational databases are especially good at answering spontaneous and unplanned inquiries. The following are all questions that could be answered easily by dBASE II commands (assuming, of course, that files containing the relevant data had been established). For most, a one-line command would suffice to select the data needed, data that could then be listed on the screen or the printer or both. If the question needed to be asked as part of a larger process, a program could be written in dBASE II and the step then performed by the command "DO program" where "program" is the name of the program.

- § What items in the pool have difficulties between .50 and .90 and a discrimination of >.20?
- § What is the performance history of a given item?
- § What were the difficulties and discriminations for items used on Form 4?
- § What items are on the draft test currently under development? What is their average difficulty and discrimination?
- § What items are eligible to be used as equator items on the next test form to be developed?
- § How many items are in the pool in each content category and skill level?
- § What is the average difficulty of items in each content category? Of items written by a certain item writer?
- § Which items need to be reviewed or revised before their next use?
- § Create an index of all the technical terms used in items on Form 3.

Figure 8

Description of the Dummy Database
Constructed to Test ITEMBANK Programs

Characteristic	POOL File	HIST File
Contents	Master item data	Item usage data
Organization	One record per item	One record per use of an item
Number of records	124	280
Structure	see next page	see next page

Each item in the POOL file is represented by 1 to 5 uses in the HIST file.

The HIST file represents data from eight test forms, their numbers and dates shown below. The final test form, number 32, is not entered to the HIST file but rather functions as the "draft test," i.e, the test under development. It is used to test programs associated with the entry of items to the draft test file. The draft test file (TEST2) structure is shown after the POOL and HIST structures.

Administration Date	Form Number	Equator Form Numbers	
		E1	E2
80/04/19	01	--	--
80/10/25	02	--	01
81/04/18	11	01	02
81/10/17	12	02	11
82/04/17	21	11	12
82/10/16	22	12	21
83/04/16	31	21	22
83/10/15	32	22	31

The composition of each 40-item test form, broken down by content classification and function, is shown below. "New" items refer to both NN- and NR-type items.

Content Classification	Function				
	E1	E2	U-	New	Total
I	2	2	2	2	8
II	3	3	3	3	12
III	2	2	2	2	8
IV	3	3	3	3	12
Total	10	10	10	10	40

In the ITEMBANK system, programs and dBASE II itself are on the A disk and the database and memory files are on the B disk.

See Appendix A for a complete listing of the POOL and HIST files.

```
.
. use b:pool
. list structure
STRUCTURE FOR FILE:   B:POOL     .DBF
NUMBER OF RECORDS:    00124
DATE OF LAST UPDATE: 04/16/84
PRIMARY USE DATABASE
FLD        NAME        TYPE WIDTH    DEC
001       ID            C     004
002       MKEY          C     001
003       CONT          C     001
004       FMT           C     002
005       LINES         C     002
006       STATUS        C     002
007       GRAF          C     004
008       F1            C     001
009       F2            C     001
** TOTAL **              000:9
.
.
.
.
.
.
.
. use b:hist
. list structure
STRUCTURE FOR FILE:   B:HIST     .DBF
NUMBER OF RECORDS:    00280
DATE OF LAST UPDATE: 04/16/84
PRIMARY USE DATABASE
FLD        NAME        TYPE WIDTH    DEC
001       ID            C     004
002       DATE          C     008
003       FORM          C     002
004       NUM           C     002
005       FKEY          C     001
006       FUNC          C     002
007       DIFF          N     002
008       DISC          N     002
009       F3            C     001
010       F4            C     001
** TOTAL **              00026
.
.
.
.
.
```

```
.
.
.
. use b:test2
. list structure
STRUCTURE FOR FILE:   B:TEST2    .DBF
NUMBER OF RECORDS:    00032
DATE OF LAST UPDATE:  04/17/84
PRIMARY USE DATABASE
FLD        NAME        TYPE  WIDTH    DEC
001        ID           C     004
002        MKEY         C     001
003        CONT         C     001
004        FMT          C     002
005        LINES        C     002
006        GRAF         C     004
007        F1           C     001
008        F2           C     001
009        FORM         C     002
010        NUM          C     002
011        DIFF         N     002
012        DISC         N     002
013        F3           C     001
014        F4           C     001
015        TBOOK        C     003
016        TREPL        C     003
017        TDATE        C     008
018        TFORM        C     002
019        TNUM         C     002
020        TFKEY        C     001
021        TFUNC        C     002
022        TDIFF        N     002
023        TDISC        N     002
** TOTAL **                00052
.
.
.
.
```

Figure 9

POOL File Structure

The POOL file contains one record per item, each item identified by a unique ID number. The field names, variable types, and variable lengths shown here are, with two exceptions, those used in the dummy database used to test the ITEMBANK programs. The exceptions, the SET and SKILL fields, are recommended for inclusion in pool files but were impractical to use in the dummy database because of the short length of the tests that were used to construct it. In the Variable Type/Length column below, the letters represent the three dBASE II variable types: C = character (i.e., alphanumeric) variables, N = numeric variables, and L = logical variables.

Field Name	Variable Type/Length	Description
ID	C/4	Item ID number. A number that uniquely identifies the item. It is useful to have the number correspond to the physical location of the item text. In the dummy database, it is entered as three numbers separated by a dash in the form n-nn, the numbers indicating the physical location of the item in a word processor system. For example, item number 3-45 is on disk 3, segment 45. In database jargon, such a unique identifier is called the primary key. If a database does not permit this arrangement, then it would be wise to have another field to indicate the location of the item text for use by word processor personnel.
MKEY	C/1	Master key. The position of the key (correct answer) as the item is stored in the word processor. The dummy database assumes four-choice items and indicates the position of the key with letters: A, B, C, D. Numbers could have been used instead of letters. This key position may change when the item is actually used on a test (see the FKEY field in the HIST file).
CONT	C/1	Content. The content classification tested by the item. Although only one field is used for this data in the dummy database, a real-world database may need to use several such fields to indicate the division of the content into major areas, topics, subtopics, etc. (In practice, some tests use as many as six content fields.) It is represented in the dummy database as a single-digit Arabic numeral although all ITEMBANK programs that print it use a small DO CASE subroutine to convert it to Roman numerals. Using the Arabic numeral in the database makes it easier to sort and index on the field and takes up less space. However, people are often more comfortable using Roman numerals for at least the

first major content level, especially if such use was established long before the database was converted to a computer-based file. The dummy database assumes four content areas.

SKILL	C/1	Cognitive Skill. The cognitive skill tested by the item, e.g., recall, knowledge, understanding, application, etc. It may be coded as a letter or a number.
FMT	C/2	Item Format or Type. Some tests use item formats other than the traditional multiple-choice type. The fictional test associated with the dummy database uses two types, multiple-choice and option, coded MC and OP, respectively. Other types could also be coded if used on the test, e.g, essay, matching, true-false, etc.
SET	C/5	Position in an Item Set. Indicates whether the item is part of a set and its position in that set. Items in a set are based on the same stimulus material (e.g., a reading passage, a graph, a description) and cannot be used singly. This field is coded in the form mm/nn where mm is the item's position in the set and nn is the total number of items in the set. For example, the fifth item in a 10-item set would be coded 05/10. By adding a slash and an additional subfield, a code describing the nature of the stimulus material could also be entered in this field. Items that can be used singly have only blanks in this field.
LINES	C/2	Lines. The number of lines occupied by the item on the printed test. This information is useful in assembling and ordering the items and arranging them on the test pages. It is entered as a character field in the dummy database. However, if this data is to be used in any calculations, it would probably be better to enter it as a numeric field.
GRAF	C/4	Graphics. The nature and length of graphics, if any, used in this item. It is coded in the form a/nn where a is nature of the graphic and nn the number of printed lines the graphic occupies. The dummy database uses three types of graphics: P = photographs, D = line drawings, and H = any special characters that are drawn or pasted in by hand. These latter graphics usually have a line length of 00 (zero) since they are typically embedded in the printed text. They are entered into the database as a reminder to the person assembling the test that this task needs to be done. Additional codes could be used in this field to indicate other types of graphics (e.g., X-rays, slides). In a real-world

		database with many types of graphics to code, a two-character subfield might be more useful. Line length should be computed as the number of blank lines that need to be left in the item text so the graphic can be pasted in. If needed, another slash and subfield could be added to indicate the graphic's width.
STATUS	C/2	Status. The item's activity status in the pool. The dummy database uses three status codes: AC = currently active, IN = inactive, and AR = active but revised since last use. The text of AR-type items has been revised so that they must be treated as new items. Although they have been used on past forms of the test, the revisions make it likely that their performance characteristics (difficulty and discrimination) will change. AR-type items can function only as NR-type items (i.e., new due to revisions) on the test under development. No item record is ever deleted; it is simply marked inactive. This permits the reconstruction of previous forms of the test on which the item may have been used. Other codes could also be used here to indicate, for example, temporarily inactive, active but needs to be revised before being used again, etc.
F1	C/1	Flag Field 1. Used in the test development process to indicate an item's current status with respect to its selection for possible inclusion in a test under development. In the dummy database and its associated programs, status is shown in the following manner: Items selected by test development staff for a draft test are coded 1; items selected as alternate items are coded 2. If the draft test item is accepted by reviewers, it retains its code of 1; if it is rejected, it is coded 3. If an alternate item is not selected by reviewers, it retains its code of 2; if is used to substitute for a rejected item, it is coded 4. This field is redundant with field F3 in the HIST file.
F2	C/1	Flag Field 2. Used in the test development process to indicate an item's current status with respect to its function in a test under development. In the dummy database and its associated programs, this field is coded as follows: 1 = E1 items, 2 = E2 items, 3 = U- items, and 4 = NR items. See the FUNC field in the HIST file for further explanation of item functions. This field is redundant with field F4 in the HIST file.

*** Other Possible Fields ***

WRITER	C/3	Item Writer (or Source). A three-letter or three-number code indicating the writer of the item (or other item source as appropriate). Possibly useful to ensure that not too many items by one writer are used on one test form, to determine which writers develop the most successful items, or to indicate the most appropriate person to call to answer a technical question about the item's content.
REF	C/3	Reference. A three-letter or three-number code to indicate a book or journal reference for the item. Some highly technical tests require such references. Another file could be established with a primary key of the three-character code, making it possible to produce a listing of, for example, all the references used in the item pool or all references used on items in a given test form.
SUBSCALE	C/2	Subscale Membership. The item's membership on a scored subscale if this is different from the CONT or SKILL fields above.

Figure 10

HISTory File Structure

The HISTory file contains one record for each use of an item. The primary keys for this file, i.e., the keys that together make each record unique, are the item ID number and the test form number. The fields shown here are the ones used in the dummy database devised to test the ITEMBANK programs.

Field Name	Variable Type/Length	Description
ID	C/4	Item ID Number. Same as the field of the same name in the POOL file. With the test form number, it serves as the primary key for the HIST file and functions to link the POOL and HIST files together. For example, the dBASE II command JOIN will join together two files (a primary and a secondary) but requires an identical key field in each file so that a record in one may be joined to the corresponding record(s) in the other.
DATE	C/8	Administration Date. The date this test form was administered. It is coded in the form yy/mm/dd. The slashes are used here for clarity but could as easily have been omitted. If the database is large, omitting such slashes could be a significant space savings. Slashes can be inserted with a formatting command when the date is printed. The year is entered first. Sorting or indexing on the DATE field may not yield correct results if the year is put in the final position. (Actually, sorting and indexing can be done with the year in the last position but the command is a complicated one involving breaking the date into substrings. Putting the year first is easier.) In constructing the dummy database, it was assumed that the test was administered twice a year, in April and October.
FORM	C/2	Form Number. The number of the test form. With the item ID number, it forms the primary key for the HIST file. Each test form is assigned a unique number based on the year of administration and the sequence of its administration within the year. In the dummy database, the form number was coded as a two-digit number, the first digit being the last digit of the calendar year in which the test form was administered, and the second digit being either a 1 or a 2 to indicate whether it was the first or second administration of the year. For example, Form 21 was the first test administration in 1982 and Form 32 was the second administration in 1983. Notice that this causes each successive administration of the test to get a higher test form number.

Thus, sorting or indexing on FORM results in earlier forms being listed before later forms.

NOTE: For the dummy database, DATE and FORM are redundant, i.e, each form number is associated with one, and only one, administration date. Such redundancy violates the principles used in developing a third normal form. Technically, DATE should have been omitted and a third file established that listed form numbers and their respective administration dates. The redundancy was purposely built in here since, although dBASE II can have any number of data files, it can relate only two at a time. In some programs, switching from the POOL and HIST files to a third file of forms and dates would have made the programs run too slowly. However, in larger databases, the trade-off between speed and the amount of space occupied by the files might make such a form-date file worthwhile. For example, in a database of 1,000 records, removing a redundant date field would save almost 8,000 bytes.

NUM C/2 Sequence Number. The number of the item on the test form. Items are generally ordered randomly, rather than by content classification and skill. Including this field makes it possible to reconstruct a test with the items in proper order.

FKEY C/1 Form Key. The position of the key (correct answer) as the item was used on the form. As with the POOL file, the dummy database assumes four-choice items and indicates the position of the key with letters: A, B, C, D. The form key may differ from the master key because it is necessary to balance the key of the test form. If forty four-choice items are selected for a test, ten each should be keyed A, B, C, and D. However, if 15 A's and only 5 B's are selected, the key position of some of the B-keyed items will need to be shifted to balance the test.

FUNC C/2 Function. An items's function on this test form. There are five function codes: E1 = items from equator form 1, E2 = items from equator form 2, U- = items previously used on other forms, NR = items previously used but revised to the extent that they must be treated as new items, and NN = new items making their first appearance on this test form. Test forms are balanced so that the test comprises approximately 25 percent each E1, E2, U-, and new (NN, NR) items. It was assumed that the test described by the dummy database had to be horizontally equated from one administration to the next, i.e, any given scaled score resulting from the test had to represent the same level of ability from one

administration to the next regardless of any shift in the average difficulty level of the items or in the ability of the group of persons tested. The practice of equating forms requires that items of known difficulty be repeated from one form to another. Typically, these equator items are drawn from two previous test forms since this provides a more stable equating procedure. Equator items from the previous test forms are coded E1 and E2, respectively, in the dummy database. An item whose STATUS is coded AR in the POOL file is available for selection only as an NR-type item on a draft test.

DIFF	N/2	Item Difficulty. The item's difficulty on this form of the test. Coded as a two-digit number representing the percent of test takers who answered the item correctly. Like DISC below, this is a number, rather than a character, field since it is necessary to use it in computations such as averages. Also like DISC, its leading decimal point is omitted.
DISC	N/2	Item Discrimination. The item's discrimination on this form of the test. Coded as a two-digit number representing the point-biserial correlation between score on the item (i.e., right or wrong) and raw score on the test. Two discrimination fields could have been used here so as to include both biserial and point-biserial discrimination.
F3	C/1	Flag Field 3. Repeats F1 field from POOL file.
F4	C/1	Flag Field 4. Repeats F2 field from POOL file.

The redundancy between the F1 and F2 fields in the POOL file and the F3 and F4 fields in the HIST file is necessary since some ITEMBANK programs (or subprograms) use only one of the two files but require knowledge of an item's status and function. This permits an increase in speed of these programs since the other file does not have to be accessed. The redundancy also functions to some extent as a safety check to ensure that items are coded properly. Any difference between F1 and F3, or between F2 and F4, indicates an error situation that must be tracked down.

*** Other Possible HIST Fields ***

RASHDIFF	C/2	Rasch Difficulty. The item's Rasch difficulty.
RASHDISC	C/2	Rasch Discrimination. The item's Rasch discrimination.
		These two fields concerning Rasch statistics could be used in addition to, or in place of, the DIFF and DISC fields described above.
RESPA RESPB RESPC RESPD OMIT	each N/2	The percent of test takers selecting each of the four foils or omitting the item. The DIFF field above represents the percent of test takers who selected the key. An alternative approach is to enter the percent of test takers who select each foil or omit the item.
POP	N/3	The number or population of test takers for this form. Useful in interpreting the preceding five fields.

Figure 11

Other Database Files Useful in an Item Bank

In addition to the POOL and HIST files just described, other files might also be valuable depending on the user's particular application. Some of these were already alluded to in the POOL and HIST descriptions.

File: DATEFORM
Primary Key: FORM (test form number)
Other Fields: DATE (test administration date), POPN (number of persons taking this form of the test).

As indicated in the discussion of the DATE and FORM fields in the HIST file, that file should be normalized to a third normal form by creating a separate file containing test form numbers and their corresponding administration dates. Administration dates for any item in the HIST file could be obtained by joining form numbers from that file to the DATEFORM file. The POPN field is useful if, as suggested in the description of the HIST file, fields are added that report the percent of test takers who selected each foil.

File: DRAFT
Primary Key: ID (item ID number)
Other Fields: Three groups of fields for each item record: (1) data from the POOL file, (2) data on the item's latest use from the HIST file, and (3) data on the item with regard to its use in the test form currently under development. This latter group includes the fields TBOOK (control or "book" number), TREPL (the control number of an alternate item that replaced the originally selected item in the draft test), TDATE (test administration date), TFORM (test form number), TNUM (the item's sequence on the test), TFKEY (key position on the test form), TFUNC (the item's function on the test form), and TDIFF and TDISC (the item's difficulty and discrimination on the form).

This is a temporary file created and used only during the development of a test. It puts data from the POOL file and the HIST file together and adds fields parallel to those in the HIST file plus two others (TBOOK and TREPL) used only during the development stages. Thus, most of the data in the file duplicates data in existing files. It is constructed in order to make the ITEMBANK programs run faster and to prevent possible accidental alteration of fields in existing files. Numerous listings, in various orders, must be made of data for items selected for a test under development. It would be very time consuming to search the entire POOL and HIST files for such data. Putting the relevant data in a separate file shortens processing time. Since some data must be entered during the development process, keeping the records in a separate file also reduces the risk of unintentional alterations to the most important files. New items that are being used for the first time on the test form being developed have their master data first entered here, not directly to the POOL file. At the end of the test development cycle, when the test has been administered and the difficulties and discriminations have been entered, a program appends the appropriate data for that test form to the HIST file for each item and to the POOL file for the new items. (Programs in the ITEMBANK system use the file name TEST2 for this file.)

File Name: KEYWORDS
Primary Key: ID (item ID number)
Other Fields: WORD (technical terms or other key words)

In developing tests on highly technical content, it is sometimes desirable to determine the extent to which certain words, terms, or concepts are used within a given test (e.g., medical terms, drug names, etc.) When an item's text is finalized and reviewed, terms from it are selected for inclusion in this file. Data from this file can then be joined by means of the primary key to data in the DRAFT or HIST files to produce an "index" of technical terms used on a test form.

File Name: WRITERS
Primary Key: WID (item writer's ID number)
Other Fields: NAME (item writer's name), ADD (item writer's address), SPEC (item writer's technical speciality group)

One of the suggested fields in the POOL file was one for designating the writer of each item. If this is done, the field should also be added to the DRAFT file with other POOL item data. By joining the WRITERS file to the POOL or DRAFT files, it is thus possible to produce a list of the persons who wrote the items in the pool or on the draft test. This may prevent the use of too many items from one writer which, on a national test, could introduce a regional bias. It would also provide the name of the person to contact if a question should arise regarding the technical content of an item.

File Name: COMMENTS
Primary Key: ID (item ID number)
Other Fields: DATEIN (date comment entered), COMM1, COMM2 (text of the comments)

In reviewing an item, editors and reviewers often make comments regarding modifications that need to be made to the item, guidance on using the item, or other advice. It is valuable to preserve these comments in a separate file so they can be retrieved when items are being selected for a draft test. Although dBASE II permits a field length of 255 characters, this can be awkward to print in hard copy, especially if joined to POOL or HIST data on the ID key. Two comments fields of, say, 65 characters each might be easier to print and still allow ample room for complete statements.

File Name: NEWITEMS
Primary Key: ACC (some internal control or accession number)
Other Fields: Same fields as POOL file records, plus ACCDATE (date item was entered to the file.

The ITEMBANK system was developed with the assumption that no file was kept on new, unused items. The master data on an item was not developed and entered until the item had been used on at least one test form. However, some users may prefer to maintain data on such items. There are two ways to do this. One is simply to enter the item master data to the POOL file and code them NN in the STATUS field to indicate that they were new, never-before-used items. One of the advantages of a relational database is such items need not have been used on any form to be entered to the POOL file. A second approach is to maintain a separate file of data on new items. This approach would be partic-

ularly valuable if, as is often the case, a full determination of some of the item's characteristics (e.g., content classification, skill, format, lines) had not yet been made. Another consideration is the number of available new items. Adding the records on many new items to the POOL file could considerably increase the execution time for programs that used the file. Any item in this file could have its data appended first to the DRAFT file during the test development process and later, after the test had been administered, to the POOL file. The ACCDATE field is used to judge when the item was received and thus is a measure of its age. Items not used within a certain period of time after being received may need to be reviewed to determine if they are still viable and up to date with knowledge in that content field.

Figure 12

Considerations in the Development of the
ITEMBANK System Using dBASE II

1. The software (i.e., dBASE II and the programs written using it) had to be integrated with the total test development process. The system should not be simply a computerized card file of item data. Wherever possible, the machine should copy data and files, print reports, update files, etc.

2. The system should be "user-friendly." Wherever possible, clerical personnel should do data entry and maintenance. However, such personnel may not have had experience with computer data entry and may also have some uncertainties about dealing with computers. Thus, it is important that this process be as simple and error-proof as possible. This is accomplished with prepared programs accessed through menus that allow actions to be accomplished with a minimum of computer skills. The use of such menus along with built-in error checks also makes the input, processing, and output of data more uniform and less subject to errors due to individual differences in system users, whether clerical or professional staff. The system should have the capacity to permit users to develop the test on their schedule and to make modifications easily to any data previously entered.

3. The system must be flexible enough to be able to access and organize different files. One of the greatest advantages of dBASE II is that it can allow for many files, joining them as needed and use one to organize data in another. Other files (e.g., command files) contain programs for processing or instructions and formats for printing data.

Figure 13

The Four Stages of Test Development in the
ITEMBANK System:
An Overview

The first menu displayed after the ITEMBANK system shows an identifying screen, initializes the environment, and asks the user for the current stage of test development. A secondary menu can then be displayed for each stage.

1. The Draft Test Stage. This stage is concerned with producing a draft of the examination, i.e., a set of items selected by professional staff and presented to reviewers (e.g., client representatives, a steering committee of content experts, etc.). Presented with this draft test is a group of alternate items that may be substituted for any item in the draft test that is rejected by the reviewers. The stage begins with the listing of all items in the pool that are eligible for selection (the Test Drafting Kit) and ends with the review of the draft test.

2. The Working Test Stage. Following the client review, the draft test becomes a working test. Changes recommended by reviewers are incorporated into the draft examination beginning with the substitution of alternate items selected by reviewers to replace rejected items from the draft test. Data on new items not previously used on any test form are entered to the database. Programs in this stage assist in the assembly and revision of the test. The stage ends when the test, with the items in test order, is sent to reviewers for further comments and changes.

3. The Final Test Stage. Once comments are received from reviewers, the working test becomes a final test. Recommended changes, possibly including further substitution of alternate items, key changes, etc., are incorporated. The step ends when the revised test is sent to the printer and then to operations personnel for administration.

4. The Test Analysis and Itembank Updating Stage. After the test has been administered and the results analyzed, the new data on the usage of each item must be entered to the database. This involves entering item performance data (difficulty and discrimination) and appending the item data to the appropriate files: the usage data to the HIST file and the master data on new items to the POOL file. Data used only during the test development process is then deleted. Once completed, the test development cycle can begin again for the next test form.

Figure 14

The Test Development Process in the
ITEMBANK System

1. The Draft Test Stage.

A. Printing a Test Drafting Kit. To assist professional staff in the initial selection of items for a draft test, the master and history data on each item are printed out. It does this by joining together data from the POOL and HIST files. This report, called the Test Drafting Kit, is organized in four parts, listing items eligible for selection as E1 items, E2 items, U-items, and NR items, respectively. (A sample of the printout is given in Figure 17). Within each part, items are ordered by content area and by item ID number. If a COMMENTS file existed (it does not in the dummy database), then comments might also be printed at this time to alert staff to any problems with a particular item.

B. Selecting Items for the Draft Test.

C. Selecting Items for the Alternates List. Test development staff use the kit in combination with the item text printed by the word processor to make initial selections of draft test items and alternate items.

D. Creating a Draft Test. The ID numbers for the items selected for the draft test are entered to the system and a draft test file (called TEST2 in ITEMBANK) is created. This file will exist only during the test development process and will be deleted after the test is administered and the new usage data is entered to the HIST file. Entry of each item ID number causes a record to be appended to the draft test file keyed on that number. Each record contains master data on the item copied from the POOL file, the latest usage data on the item copied from the HIST file, and series of fields that will be used to handle data on the item during the test development process. This file is set up because data regarding items on a test under development must be indexed, sorted, printed and handled in other ways. Programs would run too slowly if such data had to be extracted from the POOL and HIST files each time it was needed. Putting the data in a separate file also makes operations smoother and less open to error. However, items in the POOL and HIST files are "flagged" to indicate their use and function on a test being developed.

E. Creating an Alternate Items List. The ID numbers of the items selected as alternates are entered to the system. Items selected as alternates are simply "flagged" in the flag fields. No separate file is created for them since they are listed but not usually handled in any other way until they are selected as substitutions in the draft test.

F. Changing the Draft Test or Alternate Items List. For whatever reason, test development staff may wish to make changes to items selected for the draft test or the alternate items list. Some items may be added or others deleted. An alternate item may become a draft test item and vice versa. The ITEMBANK system fully supports such changes and updates all files accordingly when they are entered.

G. Entering Book Numbers for Draft Test Items. Once the draft test is complete, or nearly complete, items need to be assigned a control number, usually called a "book" number. It is necessary to have some number to refer to the item during the test development process that reflects its order of review on the draft test. The item ID number cannot be used for this since it is random with respect to item content, skill, and function, and its sequence number in the test will not be assigned until the next stage. This book number represents the order of the item as it is presented to the reviewers. Since items are presented in order by content and function, book numbers are assigned in this order. ITEMBANK presents a screen that permits entry or change of book numbers and automatically adds this data to the draft test file. The program that does this can also be used to browse through the draft test data.

H. Printing the Test Summary Sheet for the Draft Test. A summary of the draft test item data can be printed, organized by ID number, book number, or content and function. Such summary sheets are printed in every stage with slight variations in the data provided. They function as an important element of the quality control process. This program also prints a summary of the test statistics (based on the items' latest use), that includes a count of items and their average difficulty and discrimination broken down by content classification and function. A sample printout is given in Figure 18.

I. Printing the Alternate Items List. A dBASE II utility (REPORT) is used to print basic data regarding the alternate items. Data from the POOL and HIST files are joined to create a temporary file which is then printed and deleted. A sample printout is given in Figure 19.

2. The Working Test Stage.

A. Entering Alternate Items to the Working Test. After the draft test is reviewed, it becomes known as a working test. The working test file must be updated to reflect the item substitutions made by reviewers. Rejected draft test items must be flagged in the POOL, HIST, and TEST2 files, as must the alternate items substituted for them.

B. Entering New Items to the Working Test. New items (i.e, NN-type items that have not previously been used on any test form) present a special problem since, prior to being entered to the working test file, they must receive a final editing, be entered to the word processor system, and be assigned a unique ID number. Once this is done, their master data is entered to the working test file (not the POOL file).

C. Printing a Test Assembly Guide. The editorial and quality control functions must be accomplished during this stage, can be assisted by the printing of a Test Assembly Guide. This is similar to the Test Summary Sheet but includes item line lengths and graphics data. Items can be printed in ID number order so word processor personnel can output the appropriate items, and by book number or function. Data on line length and graphics is needed in planning item arrangement on test pages and placing items in test order.

D. Entering Sequence (Test Order) Numbers to the Working Test. Once the order of the items is tentatively established, the sequence, or test order, numbers can be entered to the working test file. Like the book numbers above,

these can be changed at any time during the process, should the arrangement of the items need to be modified.

E. Balancing the Working Test Key. It may be necessary to change the position of the key (correct answer) on some items in order to balance the key for the test as a whole. If too many items are keyed A and too few B, some A's must have their keys moved to the B position in the foils. Once this is accomplished, the new key positions are entered to the working test file.

F. Printing a Test Summary Sheet. This step is essentially the same as the one of the same name in the Draft Test Stage. However, in this step, the additional option exists to print item data in test order. Several copies of the sheet may need to be printed to reflect changes to the test during this stage. A modified version of the sheet is printed for reviewers and sent to them along with a review copy of the test.

3. The Final Test Stage.

A. Making Changes to the Final Test. Once the working test has been reviewed, it is known as the final test. Changes may need to be made to the test based on reviewers' comments. These may include further item substitution, or key or content changes. Programs in this stage permit the change of any relevant data in the final test file.

B. Printing a Scoring Sheet. A scoring sheet is used in scoring the examination. It contains only the item test order numbers and the keys.

C. Printing a Test Summary Sheet. This step is essentially the same as the one of the same name in the Working Test Stage.

4. Updating and Analysis Stage.

A. Entering Item Analysis Data. Item analysis data resulting from the administration of the final test is entered to the final test file (not directly to the HIST file).

B. Printing a Final Test Summary Sheet. The Final Test Summary Sheet is the final word on this form of the test. It presents data similar to that of previous Test Summary Sheets, but the data now includes the difficulty and discrimination from this test form.

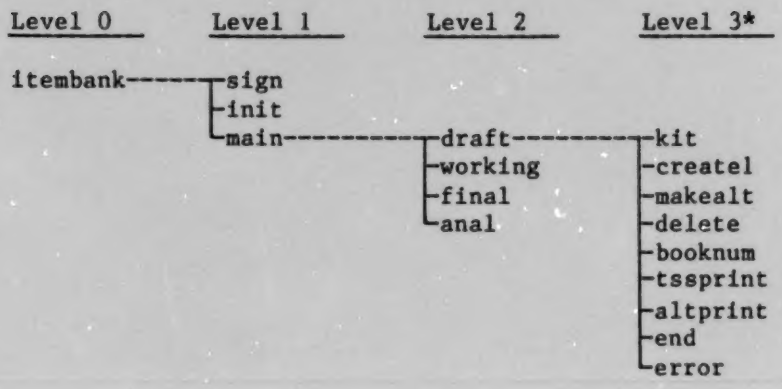
C. Updating the POOL and HIST Files. The POOL and HIST files must be updated to include data entered as a result of the administration of the test and the completion of the test development cycle. Data on NN-type items must be added to the POOL file. Usage data for each item for this form must be added to the HIST file. All flags in the two files must be brought back to zero in preparation for the beginning of the next test development cycle.

D. Cleaning Up. Additional work may need to be done on the files. If a COMMENTS file exists (it does not in the ITEMBANK dummy database), it may need to be purged of outdated comments and have current ones entered. Based on their performance on the test, some items may need to be flagged for review. Items that will be revised before their next use need to have their status changed to AR (active item but revised and thus must be treated as a new item

in its next use). Finally, the TEST2 file must be deleted.

Figure 15

ITEMBANK Main Menu Program Branching



*Level 3 is shown in more detail in the following figure.

Figure 16

Draft Stage Menu Program Branching

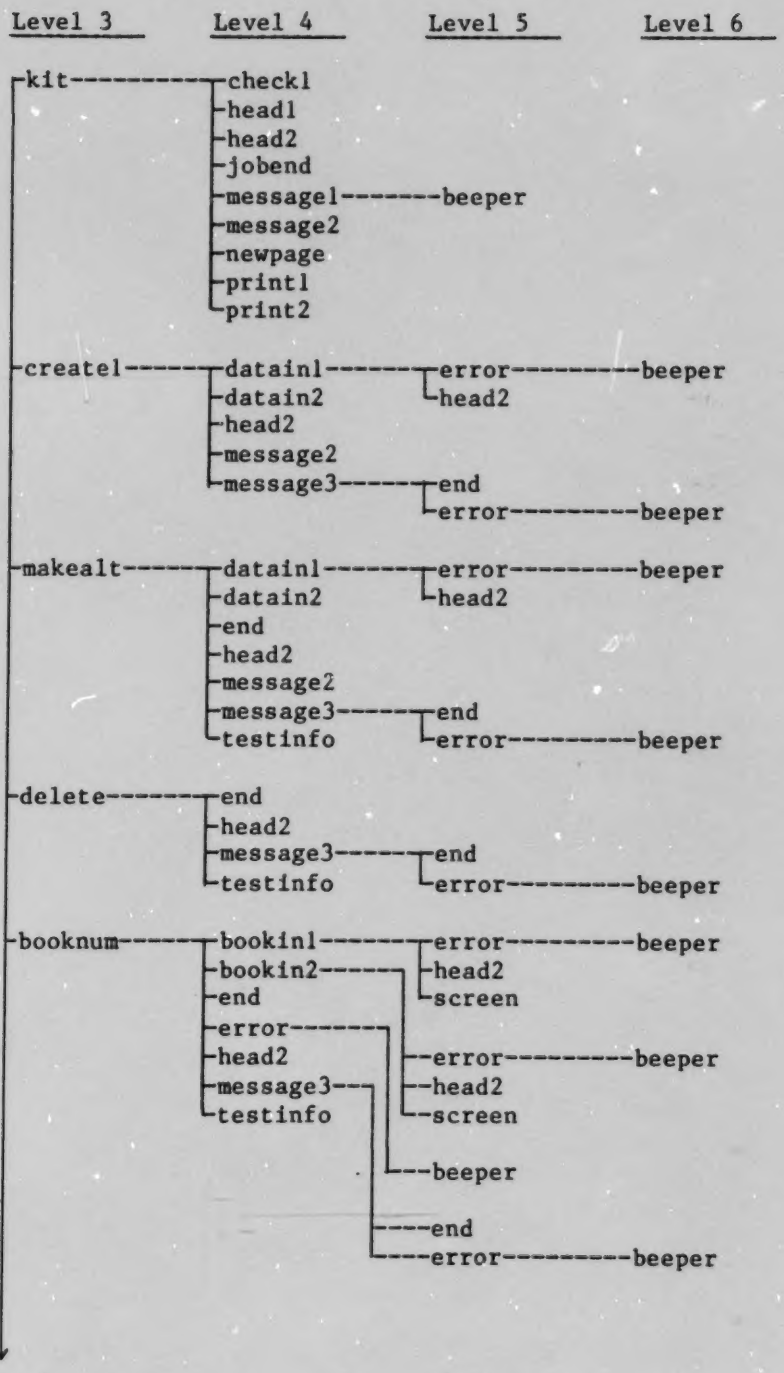


Figure 16 (Continued)

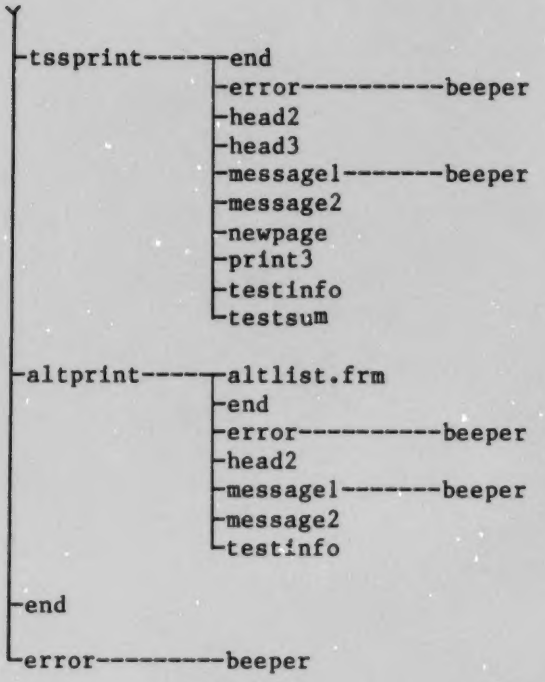


Figure 17

Sample Printout of a Page from the
ITEMBANK Test Drafting Kit

The Test Drafting Kit presents the more important master and usage data on each item in the item bank. This is only the first page of printout that might be used prior to developing test form number 32. The complete printout is 9 pages long.

ITEMBANK Test Drafting Kit - Part 1
Items available for selection as E1 items
printed 04/04/84

ID #	Mas Key	For mat	Con tent	Form	Num	Form Key	Func tion	Diff	Disc
2-10	C	MC	I	01	26	C	U-	64	73
				22	19	C	U-	69	67
				31	05	A	E2	72	65
2-11	D	MC	I	02	29	D	NN	86	41
				11	29	D	E2	85	56
				22	14	D	NR	87	62
3-02	C	MC	I	01	05	C	NN	72	40
				11	09	C	E1	72	39
				21	06	C	E1	66	27
				22	22	C	E2	76	26
3-07	A	OP	I	01	29	A	E2	83	34
				12	25	A	U-	78	34
				22	29	A	E1	88	45
3-17	C	MC	I	01	09	C	U-	81	38
				02	16	C	E2	68	44
				12	07	C	E1	74	41
				22	11	C	E1	69	40
				31	30	C	E2	72	43
3-25	B	OP	I	22	25	B	U-	73	6
1-08	A	MC	II	11	19	A	U-	88	69
				22	02	A	U-	82	61
1-26	A	OP	II	22	38	A	NN	52	47
1-34	A	MC	II	22	36	D	NN	75	18
2-14	D	MC	II	02	18	D	NN	82	49
				12	13	D	E1	80	50
				22	33	D	E1	79	57
2-26	D	OP	II	02	32	D	U-	64	22
				11	39	D	NR	41	28
				21	18	D	E1	48	25
				22	18	D	E2	49	23
2-37	C	MC	II	21	25	C	NN	79	57
				22	24	C	E2	78	50
				31	09	C	E1	76	49
2-39	D	MC	II	22	26	D	NN	61	40
3-13	B	OP	II	01	18	B	U-	60	34
				11	12	B	E1	63	37
				12	28	D	E2	59	27
				22	10	B	E1	60	35

Figure 18

Sample Printouts of a Test Summary Sheet
from the
ITEMBANK Draft Test Stage

Some columns (e.g., the test sequence number and the form key for the draft test) are blank since this data will not be supplied until a later stage. Three versions, each available at the user's choice, are shown here. These versions show items sorted by ID number, by content and function, and by book number. Later stages also provide the option of printing item data in test sequence number order. Only one copy of the summary statistics is displayed here since it is the same for all three versions.

ITEMBANK Test Summary Sheet
Form 32/Administration Date 83/10/15
printed 04/15/84

Test Development Data					Master Item Data				Latest Usage Data			
Book Num	Repl By	Test Seq#	Form Key	Function	Item ID #	Mas Key	Content	Format	Form	Num	Diff	Disc
109				E1	1-08	A	II	MC	22	02	82	61
132				E2	1-09	D	IV	MC	31	07	74	59
112				E2	1-16	D	II	MC	31	32	80	59
118				NR	1-21	B	II	MC	31	23	41	17
110				E1	1-26	A	II	OP	22	38	52	47
129				E1	1-30	B	IV	MC	22	04	48	34
111				E1	1-34	A	II	MC	22	36	75	18
133				E2	1-40	D	IV	OP	31	04	70	48
113				E2	1-43	C	II	MC	31	06	80	60
134				E2	1-48	D	IV	MC	31	11	66	72
115				U-	2-06	C	II	OP	11	32	69	60
135				U-	2-20	D	IV	MC	12	30	58	47
103				E2	2-21	A	I	MC	31	17	82	38
105				U-	2-23	B	I	MC	11	02	93	36
121				E1	2-27	C	III	OP	22	21	59	35
116				U-	2-32	B	II	MC	12	08	85	24
130				E1	2-35	A	IV	OP	22	16	89	42
125				U-	2-40	A	III	MC	02	15	77	49
122				E1	2-41	B	III	MC	22	17	79	39
131				E1	2-45	D	IV	MC	22	09	56	43
101				E1	3-02	C	I	MC	22	22	76	26
106				U-	3-11	B	I	MC	02	12	52	32
107				NR	3-18	D	I	MC	12	03	90	31
123				E2	3-19	C	III	MC	31	15	71	44
126				U-	3-22	D	III	MC	01	36	71	40
102				E1	3-25	B	I	OP	22	25	73	6
124				E2	3-28	A	III	MC	31	28	73	46
136				U-	3-29	A	IV	OP	12	19	70	23
114				E2	3-30	D	II	MC	31	16	59	28
117				U-	3-34	A	II	MC	12	09	58	15
137				U-	3-36	A	IV	MC	02	13	60	28
104				E2	3-43	C	I	MC	31	12	59	24

ITEMBANK Test Summary Sheet
Form 32/Administration Date 83/10/15
printed 04/15/84

Test Development Data					Master Item Data				Latest Usage Data			
Book Num	Repl By	Test Seq#	Form Key	Function	Item ID #	Mas Key	Content	Format	Form	Num	Diff	Disc
101				E1	3-02	C	I	MC	22	22	76	26
102				E1	3-25	B	I	OP	22	25	73	6
103				E2	2-21	A	I	MC	31	17	82	38
104				E2	3-43	C	I	MC	31	12	59	24
105				U-	2-23	B	I	MC	11	02	93	36
106				U-	3-11	B	I	MC	02	12	52	32
107				NR	3-18	D	I	MC	12	03	90	31
109				E1	1-08	A	II	MC	22	02	82	61
110				E1	1-26	A	II	OP	22	38	52	47
111				E1	1-34	A	II	MC	22	36	75	18
112				E2	1-16	D	II	MC	31	32	80	59
113				E2	1-43	C	II	MC	31	06	80	60
114				E2	3-30	D	II	MC	31	16	59	28
115				U-	2-06	C	II	OP	11	32	69	60
116				U-	2-32	B	II	MC	12	08	85	24
117				U-	3-34	A	II	MC	12	09	58	15
118				NR	1-21	B	II	MC	31	23	41	17
121				E1	2-27	C	III	OP	22	21	59	35
122				E1	2-41	B	III	MC	22	17	79	39
123				E2	3-19	C	III	MC	31	15	71	44
124				E2	3-28	A	III	MC	31	28	73	46
125				U-	2-40	A	III	MC	02	15	77	49
126				U-	3-22	D	III	MC	01	36	71	40
129				E1	1-30	B	IV	MC	22	04	48	34
130				E1	2-35	A	IV	OP	22	16	89	42
131				E1	2-45	D	IV	MC	22	09	56	43
132				E2	1-09	D	IV	MC	31	07	74	59
133				E2	1-40	D	IV	OP	31	04	70	48
134				E2	1-48	D	IV	MC	31	11	66	72
135				U-	2-20	D	IV	MC	12	30	58	47
136				U-	3-29	A	IV	OP	12	19	70	23
137				U-	3-36	A	IV	MC	02	13	60	28

ITEMBANK Test Summary Sheet
Form 32/Administration Date 83/10/15
printed 04/15/84

Test Development Data					Master Item Data				Latest Usage Data			
Book Num	Repl By	Test Seq#	Form Key	Function	Item ID #	Mas Key	Content	Format	Form	Num	Diff	Disc
101				E1	3-02	C	I	MC	22	22	76	26
102				E1	3-25	B	I	OP	22	25	73	6
109				E1	1-08	A	II	MC	22	02	82	61
110				E1	1-26	A	II	OP	22	38	52	47
111				E1	1-34	A	II	MC	22	36	75	18
121				E1	2-27	C	III	OP	22	21	59	35
122				E1	2-41	B	III	MC	22	17	79	39
129				E1	1-30	B	IV	MC	22	04	48	34
130				E1	2-35	A	IV	OP	22	16	89	42
131				E1	2-45	D	IV	MC	22	09	56	43
103				E2	2-21	A	I	MC	31	17	82	38
104				E2	3-43	C	I	MC	31	12	59	24
112				E2	1-16	D	II	MC	31	32	80	59
113				E2	1-43	C	II	MC	31	06	80	60
114				E2	3-30	D	II	MC	31	16	59	28
123				E2	3-19	C	III	MC	31	15	71	44
124				E2	3-28	A	III	MC	31	28	73	46
132				E2	1-09	D	IV	MC	31	07	74	59
133				E2	1-40	D	IV	OP	31	04	70	48
134				E2	1-48	D	IV	MC	31	11	66	72
107				NR	3-18	D	I	MC	12	03	90	31
118				NR	1-21	B	II	MC	31	23	41	17
105				U-	2-23	B	I	MC	11	02	93	36
106				U-	3-11	B	I	MC	02	12	52	32
115				U-	2-06	C	II	OP	11	32	69	60
116				U-	2-32	B	II	MC	12	08	85	24
117				U-	3-34	A	II	MC	12	09	58	15
125				U-	2-40	A	III	MC	02	15	77	49
126				U-	3-22	D	III	MC	01	36	71	40
135				U-	2-20	D	IV	MC	12	30	58	47
136				U-	3-29	A	IV	OP	12	19	70	23
137				U-	3-36	A	IV	MC	02	13	60	28

Summary of Item Statistics

Breakdown by		# of Items	Mean Diff	Mean Disc
Item Function	E1	10	68	35
	E2	10	71	71
	U-	10	69	35
	NR	2	--	--
Content Class	I	7	72	27
	II	10	71	41
	III	6	71	42
	IV	9	65	44
Total Test		32	69	39

Figure 19

Sample Printout of an Alternate Items List
from the
ITEMBANK Draft Test Stage

The user has the choice of printing the alternate items for one of the four item functions (E1, E2, U-, or NR) or of printing all items on the list. The user has the further choice of having the items printed in order by ID number, by content classification, function, and ID number (if all items are being printed), or by content classification and ID number (if only one type of item is being printed). The samples shown here are for E1 alternate items only.

Alternate Items List

Item ID #	Mas Key	Cont	For mat	Test Form	Num	Diff	Disc
3-07	A	I	OP	22	29	88	45
3-17	C	I	MC	22	11	69	40
2-11	D	I	MC	22	14	87	62
2-10	C	I	MC	22	19	69	67
2-14	D	II	MC	22	33	79	57
3-13	B	II	OP	22	10	60	35
3-21	A	II	MC	22	08	86	20
2-26	D	II	OP	22	18	49	23
2-37	C	II	MC	22	24	78	50
3-26	B	II	MC	22	37	89	29
2-39	D	II	MC	22	26	61	40
3-27	C	II	OP	22	15	84	48
1-25	B	III	MC	22	23	69	24
1-13	A	III	OP	22	30	58	66
1-39	B	III	OP	22	03	83	59
2-03	A	III	MC	22	05	69	66
3-41	C	III	MC	22	40	55	70
3-12	B	III	MC	22	31	87	44
3-31	B	IV	MC	22	27	80	34
3-38	B	IV	MC	22	39	83	25
3-15	B	IV	OP	22	34	87	40
1-37	C	IV	MC	22	13	88	33
1-41	A	IV	OP	22	32	79	62
2-42	C	IV	MC	22	12	75	49
3-06	A	IV	MC	22	06	92	36
3-24	D	IV	MC	22	35	70	21

Alternate Items List

Item ID #	Mas Key	Cont	For mat	Test Form	Num	Diff	Disc
1-13	A	III	OP	22	30	58	66
1-25	B	III	MC	22	23	69	24
1-37	C	IV	MC	22	13	88	33
1-39	B	III	OP	22	03	83	59
1-41	A	IV	OP	22	32	79	62
2-03	A	III	MC	22	05	69	66
2-10	C	I	MC	22	19	69	67
2-11	D	I	MC	22	14	87	62
2-14	D	II	MC	22	33	79	57
2-26	D	II	OP	22	18	49	23
2-37	C	II	MC	22	24	78	50
2-39	D	II	MC	22	26	61	40
2-42	C	IV	MC	22	12	75	49
3-06	A	IV	MC	22	06	92	36
3-07	A	I	OP	22	29	88	45
3-12	B	III	MC	22	31	87	44
3-13	B	II	OP	22	10	60	35
3-15	B	IV	OP	22	34	87	40
3-17	C	I	MC	22	11	69	40
3-21	A	II	MC	22	08	86	20
3-24	D	IV	MC	22	35	70	21
3-26	B	II	MC	22	37	89	29
3-27	C	II	OP	22	15	84	48
3-31	B	IV	MC	22	27	80	34
3-38	B	IV	MC	22	39	83	25
3-41	C	III	MC	22	40	55	70

Figure 20

Considerations in Writing the ITEMBANK Programs

Screen vs. Hard Copy. Entry of data and selection of various options and steps are handled with the user sitting at the terminal and viewing data and results on the CRT. However, most program output is in hard copy form (printed in this case on a dot matrix printer). This was done intentionally since it was deemed too tiring and time-consuming for a user to review data on a large number of items via the terminal screen. The review of item performance data as part of the development process to create a draft test and alternates list or the review of summary data on items in the draft test file is better conducted with a printed copy in front of the user. It is possible in the booknum program to view data on each item in the draft test file, one item at a time, but in general the programs were designed so that the user would spend as little time as possible at the terminal.

Use of Prompts and Menus. To ensure the accuracy and uniformity of data entry, the ITEMBANK programs make extensive use of menus and prompts to the user. Menus permit a number of complex operations to be performed with the press of only one key. All programs allow the user the option of returning to the stage menu or the main menu. Prompts ask the user for the entry of specific data in a specific order. Each data line is labeled accordingly and the screen area in which the data is entered is in a different color (usually magenta lettering on a blue background) from the rest of the screen.

Data Verification and Error Checks. The ITEMBANK programs further ensure the accuracy and uniformity of data through the use of a number of verification procedures. The user must press return after entering each data element (i.e., the dBASE II command SET CONFIRM ON is issued in initializing the system). After the last data element in a set is entered, the data is checked for blank values (which will usually terminate data entry for that step), and the data is then redisplayed as it will be entered to the file. If any lower case letters have been entered, they are converted to upper case. A prompt then appears that asks the user to verify the correctness of the data, usually by pressing "V", although the booknum program has other verifications that can be entered. Pressing any other key cancels the entry and permits data entry to continue. Some entered data is also checked and verified by the system, e.g., the existence of an item ID number, whether test form numbers are valid and have been entered in the correct order, etc.

Feedback. The user receives a number of feedback messages containing information about the current functioning of the programs. The user is told when files are being processed or data being entered or deleted from a file. The user also receives messages regarding possible errors or problems, e.g., the entry of a non-existent item ID number or test form number, the need to check that the printer is turned on, etc. Many of the steps in the Draft Stage begin by reminding the user of the form number, administration date, and equator form numbers of the draft test currently under development.

Program Speed and Organization. Programs were designed to run as quickly as possible consistent with an economical use of the user's time. It was discovered in testing a number of draft programs that the program with the fastest overall time for performing a given step might be frustrating for users

because it tried to do too much after the entry of each data element. For example, the createl program, which creates a draft test file or adds item data to an existing draft test file, was originally written in such a way that the user had to spend over 1.5 minutes after entering each item ID number and function for the relevant data to be drawn from the the POOL and HIST files and added to the draft test (TEST2) file. The rewrite seen here (in Appendix B) places much of the work of adding data to the draft test file at the end of the program, after the user has indicated that data entry has been completed. Initial data entry for a draft test item ID number and function now occupies only few seconds. When the user indicates there is no more data to enter, a message appears to indicate that files are being processed. The user may then leave the terminal and do other things while the program proceeds to complete the file processing.

Let The System Do The Work. Data entry and manipulation by the user was minimized wherever possible. For example, in creating the draft test file, the user enters only the minumum of information: each item's ID number and its function on the test under development. All other data needed to create the draft test file is drawn by the program from the existing data files and copied to the draft test file. There is no need for the user to copy and reenter data. As another example, the Test Summary Sheet may be printed with the items listed in order by ID number, book number, and so on. The user does not need to be content with only one listing order since the system will provide the data in the order desired. Later, at the end of the test development process, when item performance data is entered, the user needs to enter only the difficulty and discrimination values. The system then automatically performs all the functions to update the POOL and HIST files.

Dynamics. The programs were made as dynamic as possible by the use of color and sound. The ITEMBANK system was implemented on an IBM PC with an Amidek color monitor. Instructions and information are presented on the screen in green, warning messages in yellow, step titles in blue with cyan borders, verification instructions in cyan on a blue background, error messages in flashing red, etc. The IBM PC has a choice of 256 colors that can be called up using the dBASE II command "SET COLOR TO mm,nn" where mm is the color code for GET statements (i.e., data being entered) and nn the color code for SAY statements (i.e., the regular screen display). There are many color set statements throughout the programs given in Appendix B. It is also possible to ring the bell to draw particular attention to especially urgent messages. See the beeper program for an example.

References

General Sources of Information on dBASE II

Further information on dBASE II is available from its developers, Ashton-Tate, 10150 West Jefferson Boulevard, Culver City, CA 90230, (800) 437-4329.

Descriptions of dBASE II and its commands and functions have been purposely omitted here since there are a number of good sources of information on it. PC magazine, in its February 7, 1984 issue, printed 12 articles on dBASE II. The article by Hart in that issue is an especially good introduction to the dBASE II commands. There are too many guides and aids available for learning dBASE II to attempt to list them here. One of the articles in the PC magazine issue just mentioned has a listing and review of some of them.

The journal Data-Based Advisor deals with DBMS software for microcomputers with a special focus on dBASE II. Some of its features helped solve a number of programming problems in ITEMBANK. The journal is published monthly by Data Based Solutions, 1975 5th Avenue, Suite 105, San Diego, CA 92101. Subscriptions cost $35 per year and may be obtained from Subscription Department, Data Based Advisor, PO Box 9797, San Diego, CA 92109.

Specific Sources Used in This Paper

Codd, E. F. Relational Database: A Practical Foundation for Productivity. Communications of the ACM, February, 1982.

Date, C. J. An Introduction to Database Systems. Reading, MA: Addison-Wesley, 1976.

Kent, William. A Simple Guide to Five Normal Forms in Relational Database Theory. Communications of the ACM, February, 1983.

Martin, James. Principles of Data-Base Management. Englewood Cliffs, NJ: Prentice-Hall, 1976.

----------. Computer Data-Base Organization. Englewood Cliffs, NJ: Prentice-Hall, 1975.

Schussel, George. 4th Generation Data Management Software: Seminar Workbook. Digital Consulting Associates, Inc., no date.

APPENDIX A

LISTING OF THE ITEMBANK DUMMY DATABASE

This appendix lists all data used in the POOL and HIST files, i.e., the dummy database used to test the programs in the ITEMBANK system. Programs for the Draft Test Stage are included in Appendix B. The draft test used to test the programs (called TEST2 in the ITEMBANK programs) is included in the Test Summary Sheet (Figure 18) in the basic paper.

The contents of Appendix A are copyright © 1984 by The American College Testing Program, Inc. All rights reserved.

POOL File Structure

```
STRUCTURE FOR FILE:   B:POOL     .DBF
NUMBER OF RECORDS:    00124
DATE OF LAST UPDATE:  04/16/84
PRIMARY USE DATABASE
FLD        NAME       TYPE WIDTH    DEC
001        ID          C     004
002        MKEY        C     001
003        CONT        C     001
004        FMT         C     002
005        LINES       C     002
006        STATUS      C     002
007        GRAF        C     004
008        F1          C     001
009        F2          C     001
** TOTAL **                 00019
```

HIST File Structure

```
STRUCTURE FOR FILE:   B:HIST     .DBF
NUMBER OF RECORDS:    00280
DATE OF LAST UPDATE:  04/16/84
PRIMARY USE DATABASE
FLD        NAME       TYPE WIDTH    DEC
001        ID          C     004
002        DATE        C     008
003        FORM        C     002
004        NUM         C     002
005        FKEY        C     001
006        FUNC        C     002
007        DIFF        N     002
008        DISC        N     002
009        F3          C     001
010        F4          C     001
** TOTAL **                 00026
```

Listing of the POOL File Data

```
00001   1-15 B 1 MC 08 AC        0 0
00002   1-17 C 1 MC 09 IN        0 0
00003   2-04 D 1 MC 10 IN        0 0
00004   2-05 A 1 MC 11 AC        0 0
00005   2-07 B 1 OP 13 AC H/00   0 0
00006   2-10 C 1 MC 09 AC        0 0
00007   2-11 D 1 MC 09 AC        0 0
00008   2-21 A 1 MC 10 AC        0 0
00009   2-23 B 1 MC 32 AC D/22   0 0
00010   2-25 C 1 OP 14 AC        0 0
00011   2-38 D 1 MC 12 IN        0 0
00012   2-44 A 1 MC 08 AC        0 0
00013   2-46 B 1 MC 08 AC        0 0
00014   3-02 C 1 MC 09 AC        0 0
00015   3-04 D 1 OP 12 AC        0 0
00016   3-07 A 1 OP 13 AC        0 0
00017   3-11 B 1 MC 11 AC        0 0
00018   3-17 C 1 MC 08 AC        0 0
00019   3-18 D 1 MC 18 AR H/0C   0 0
00020   3-23 A 1 MC 09 AR        0 0
00021   3-25 B 1 OP 14 AC        0 0
00022   3-43 C 1 MC 12 AC        0 0
00023   1-08 A 2 MC 08 AC        0 0
00024   1-10 B 2 MC 09 IN        0 0
00025   1-11 C 2 OP 12 AC        0 0
00026   1-16 D 2 MC 10 AC        0 0
00027   1-18 A 2 MC 10 IN        0 0
00028   1-21 B 2 MC 27 AR P/15   0 0
00029   1-22 C 2 MC 09 AC        0 0
00030   1-24 D 2 OP 13 AC        0 0
00031   1-26 A 2 OP 14 AC        0 0
00032   1-27 B 2 MC 09 AC        0 0
00033   1-29 C 2 MC 08 AC        0 0
00034   1-31 D 2 MC 09 AC        0 0
00035   1-34 A 2 MC 09 AC        0 0
00036   1-38 B 2 OP 19 AC D/07   0 0
00037   1-43 C 2 MC 10 AC        0 0
00038   1-44 D 2 MC 08 AC        0 0
00039   1-50 A 2 MC 22 AC D/12   0 0
00040   2-02 B 2 MC 09 AC        0 0
00041   2-06 C 2 OP 13 AC        0 0
00042   2-13 A 2 MC 09 IN        0 0
00043   2-14 D 2 MC 14 AC        0 0
00044   2-19 B 2 MC 09 AC        0 0
00045   2-22 C 2 MC 09 AC        0 0
00046   2-26 D 2 OP 14 AC        0 0
00047   2-30 A 2 OP 12 AR        0 0
00048   2-32 B 2 MC 08 AC        0 0
00049   2-37 C 2 MC 13 AC D/08   0 0
00050   2-39 D 2 MC 14 AC        0 0
```

Listing of the POOL File Data

```
00051   3-05 A 2 MC 09 IN        0 0
00052   3-13 B 2 OP 13 AC        0 0
00053   3-14 C 2 MC 10 AC        0 0
00054   3-16 D 2 MC 09 AC        0 0
00055   3-21 A 2 MC 09 AC        0 0
00056   3-26 B 2 MC 09 AC        0 0
00057   3-27 C 2 OP 14 AC        0 0
00058   3-30 D 2 MC 08 AC        0 0
00059   3-34 A 2 MC 25 AC P/21   0 0
00060   3-37 B 2 MC 10 AC        0 0
00061   3-40 C 2 MC 11 AR        0 0
00062   1-03 D 3 OP 12 AC        0 0
00063   1-13 A 3 OP 13 AC        0 0
00064   1-25 B 3 MC 09 AC H/00   0 0
00065   1-33 C 3 MC 09 IN        0 0
00066   1-35 D 3 MC 10 AC        0 0
00067   1-36 A 3 MC 08 AC        0 0
00068   1-39 B 3 OP 14 AC        0 0
00069   1-45 C 3 MC 19 AC D/11   0 0
00070   1-47 D 3 MC 10 AC        0 0
00071   2-03 A 3 MC 09 AC        0 0
00072   2-09 B 3 MC 09 AC        0 0
00073   2-27 C 3 OP 13 AC        0 0
00074   2-33 D 3 MC 08 AC        0 0
00075   2-40 A 3 MC 10 AC        0 0
00076   2-41 B 3 MC 13 AC        0 0
00077   3-01 C 3 MC 09 AC        0 0
00078   3-03 D 3 OP 12 IN        0 0
00079   3-08 A 3 OP 24 IN D/12   0 0
00080   3-12 B 3 MC 08 AC        0 0
00081   3-19 C 3 MC 08 AC        0 0
00082   3-22 D 3 MC 09 AC        0 0
00083   3-28 A 3 MC 10 AC        0 0
00084   3-32 B 3 OP 12 AC H/00   0 0
00085   3-41 C 3 MC 09 AC        0 0
00086   1-09 D 4 MC 11 AC        0 0
00087   1-12 A 4 MC 12 AC        0 0
00088   1-14 B 4 MC 09 AC        0 0
00089   1-19 C 4 OP 14 AR        0 0
00090   1-23 D 4 MC 27 IN D/22   0 0
00091   1-28 A 4 MC 09 AR        0 0
00092   1-30 B 4 MC 10 AC        0 0
00093   1-37 C 4 MC 09 AC        0 0
00094   1-40 D 4 OP 13 AC        0 0
00095   1-41 A 4 OP 14 AC        0 0
00096   1-42 B 4 MC 08 AC        0 0
00097   1-46 C 4 MC 08 AC        0 0
00098   1-48 D 4 MC 09 AC        0 0
00099   1-51 B 4 OP 13 AC        0 0
00100   2-01 C 4 MC 10 AC        0 0
```

Listing of the POOL File Data

```
00101   2-12 D 4 MC 11 AC        0 0
00102   2-17 B 4 MC 09 AC        0 0
00103   2-18 C 4 OP 12 IN        0 0
00104   2-20 D 4 MC 10 AC        0 0
00105   2-24 A 4 MC 09 AC        0 0
00106   2-29 B 4 MC 09 IN        0 0
00107   2-31 C 4 MC 30 AC D/22   0 0
00108   2-34 D 4 OP 13 AC        0 0
00109   2-35 A 4 OP 15 AC        0 0
00110   2-36 B 4 MC 12 AC        0 0
00111   2-42 C 4 MC 08 AC        0 0
00112   2-45 D 4 MC 08 AC        0 0
00113   3-06 A 4 MC 09 AC        0 0
00114   3-15 B 4 OP 14 AC        0 0
00115   3-20 C 4 MC 10 AC        0 0
00116   3-24 D 4 MC 11 AC        0 0
00117   3-29 A 4 OP 25 AC D/12   0 0
00118   3-31 B 4 MC 09 AC        0 0
00119   3-33 C 4 OP 12 AC        0 0
00120   3-35 D 4 MC 09 IN        0 0
00121   3-36 A 4 MC 09 AC        0 0
00122   3-38 B 4 MC 08 AC        0 0
00123   3-39 C 4 MC 21 AC P/10   0 0
00124   3-42 D 4 OP 13 AC        0 0
```

Listing of the HIST File Data

00001	2-21	80/04/19	01	01	A	E1	76	43	0	0
00002	2-44	80/04/19	01	17	A	E1	66	49	0	0
00003	1-50	80/04/19	01	07	A	E1	58	55	0	0
00004	1-44	80/04/19	01	28	D	E1	81	52	0	0
00005	3-34	80/04/19	01	11	A	E1	62	20	0	0
00006	1-36	80/04/19	01	35	D	E1	82	51	0	0
00007	1-13	80/04/19	01	02	A	E1	61	52	0	0
00008	3-35	80/04/19	01	30	D	E1	54	25	0	0
00009	2-35	80/04/19	01	32	A	E1	90	45	0	0
00010	3-38	80/04/19	01	24	B	E1	80	31	0	0
00011	3-07	80/04/19	01	29	A	E2	83	34	0	0
00012	2-25	80/04/19	01	19	C	E2	51	54	0	0
00013	2-02	80/04/19	01	08	B	E2	79	19	0	0
00014	3-05	80/04/19	01	23	A	E2	73	48	0	0
00015	1-24	80/04/19	01	12	D	E2	45	29	0	0
00016	3-08	80/04/19	01	03	A	E2	52	31	0	0
00017	3-22	80/04/19	01	36	D	E2	71	40	0	0
00018	3-20	80/04/19	01	15	C	E2	82	33	0	0
00019	3-24	80/04/19	01	40	D	E2	67	27	0	0
00020	1-14	80/04/19	01	25	B	E2	63	56	0	0
00021	3-17	80/04/19	01	09	C	U-	81	38	0	0
00022	2-10	80/04/19	01	26	C	U-	64	73	0	0
00023	2-13	80/04/19	01	04	C	U-	71	4	0	0
00024	3-13	80/04/19	01	18	B	U-	60	34	0	0
00025	3-14	80/04/19	01	06	C	U-	89	20	0	0
00026	3-03	80/04/19	01	37	D	U-	40	22	0	0
00027	1-35	80/04/19	01	13	D	U-	87	57	0	0
00028	1-46	80/04/19	01	31	C	U-	65	52	0	0
00029	2-01	80/04/19	01	33	C	U-	72	46	0	0
00030	1-42	80/04/19	01	20	B	U-	82	58	0	0
00031	2-04	80/04/19	01	22	D	NN	68	47	0	0
00032	3-02	80/04/19	01	05	C	NN	72	40	0	0
00033	3-40	80/04/19	01	38	C	NN	72	59	0	0
00034	1-10	80/04/19	01	10	B	NN	93	51	0	0
00035	3-37	80/04/19	01	34	B	NN	50	15	0	0
00036	3-12	80/04/19	01	14	B	NN	91	41	0	0
00037	3-28	80/04/19	01	21	A	NN	65	42	0	0
00038	2-29	80/04/19	01	16	B	NN	42	45	0	0
00039	2-12	80/04/19	01	39	D	NN	64	38	0	0
00040	2-17	80/04/19	01	27	B	NN	89	47	0	0
00041	2-05	80/10/25	02	05	A	E1	69	46	0	0
00042	2-07	80/10/25	02	09	B	E1	76	25	0	0
00043	2-22	80/10/25	02	35	C	E1	69	59	0	0
00044	2-06	80/10/25	02	22	C	E1	71	63	0	0
00045	2-30	80/10/25	02	39	A	E1	54	55	0	0
00046	1-47	80/10/25	02	31	D	E1	70	57	0	0
00047	2-40	80/10/25	02	15	A	E1	77	49	0	0
00048	1-48	80/10/25	02	26	D	E1	69	58	0	0
00049	1-40	80/10/25	02	03	B	E1	66	56	0	0
00050	1-12	80/10/25	02	21	A	E1	89	69	0	0

Listing of the HIST File Data

```
00051   3-17 80/10/25 02 16 C E2   68   44 0 0
00052   2-04 80/10/25 02 02 C E2   67   45 0 0
00053   3-40 80/10/25 02 38 C E2   68   60 0 0
00054   1-50 80/10/25 02 25 A E2   59   53 0 0
00055   3-14 80/10/25 02 06 C E2   94   33 0 0
00056   3-28 80/10/25 02 36 A E2   64   65 0 0
00057   1-36 80/10/25 02 14 A E2   82   52 0 0
00058   2-12 80/10/25 02 23 D E2   62   36 0 0
00059   1-46 80/10/25 02 10 C E2   60   51 0 0
00060   1-42 80/10/25 02 30 B E2   80   57 0 0
00061   3-04 80/10/25 02 01 D U-   95   44 0 0
00062   1-15 80/10/25 02 07 B U-   51   57 0 0
00063   1-38 80/10/25 02 20 B U-   81   49 0 0
00064   2-26 80/10/25 02 32 D U-   64   22 0 0
00065   2-19 80/10/25 02 11 B U-   75   51 0 0
00066   2-27 80/10/25 02 27 C U-   56   26 0 0
00067   2-33 80/10/25 02 37 D U-   87   45 0 0
00068   2-42 80/10/25 02 17 C U-   73   45 0 0
00069   2-36 80/10/25 02 34 B U-   81   54 0 0
00070   3-36 80/10/25 02 13 A U-   60   28 0 0
00071   3-11 80/10/25 02 12 B NN   52   32 0 0
00072   2-11 80/10/25 02 29 D NN   86   41 0 0
00073   2-14 80/10/25 02 18 D NN   82   49 0 0
00074   3-05 80/10/25 02 24 A NN   73   48 0 0
00075   1-22 80/10/25 02 04 C NN   24   11 0 0
00076   3-19 80/10/25 02 28 B NN   71   42 0 0
00077   1-33 80/10/25 02 33 C NN   57   41 0 0
00078   1-28 80/10/25 02 08 A NN   63   51 0 0
00079   1-23 80/10/25 02 40 D NN   43   28 0 0
00080   3-31 80/10/25 02 19 B NN   63   46 0 0
00081   3-02 81/04/18 11 09 C E1   72   39 0 0
00082   2-21 81/04/18 11 18 A E1   78   49 0 0
00083   3-13 81/04/18 11 12 B E1   63   37 0 0
00084   2-06 81/04/18 11 32 C E1   69   60 0 0
00085   2-02 81/04/18 11 24 B E1   76   37 0 0
00086   1-13 81/04/18 11 37 A E1   63   66 0 0
00087   1-35 81/04/18 11 04 D E1   85   60 0 0
00088   2-01 81/04/18 11 11 C E1   75   57 0 0
00089   2-17 81/04/18 11 30 B E1   88   50 0 0
00090   3-35 81/04/18 11 08 D E1   45   10 0 0
00091   1-15 81/04/18 11 05 B E2   50   55 0 0
00092   2-11 81/04/18 11 29 D E2   85   56 0 0
00093   3-05 81/04/18 11 10 A E2   73   53 0 0
00094   2-19 81/04/18 11 40 D E2   72   54 0 0
00095   1-38 81/04/18 11 17 B E2   83   53 0 0
00096   3-19 81/04/18 11 23 C E2   71   37 0 0
00097   1-47 81/04/18 11 31 D E2   70   55 0 0
00098   1-48 81/04/18 11 25 D E2   76   64 0 0
00099   1-28 81/04/18 11 33 A E2   52   40 0 0
00100   2-36 81/04/18 11 03 B E2   83   47 0 0
```

Listing of the HIST File Data

```
00101  3-23 81/04/18 11 26 C U-  92  66 0 0
00102  2-38 81/04/18 11 13 D U-  60  46 0 0
00103  1-08 81/04/18 11 19 A U-  88  69 0 0
00104  1-18 81/04/18 11 01 A U-  44  46 0 0
00105  1-43 81/04/18 11 34 C U-  86  54 0 0
00106  1-03 81/04/18 11 06 D U-  93  54 0 0
00107  2-09 81/04/18 11 36 C U-  61  65 0 0
00108  1-30 81/04/18 11 28 B U-  56  45 0 0
00109  2-31 81/04/18 11 16 C U-  65  29 0 0
00110  3-06 81/04/18 11 21 A U-  94  38 0 0
00111  2-23 81/04/18 11 02 A NN  93  36 0 0
00112  3-18 81/04/18 11 15 D NN  79  39 0 0
00113  2-26 81/04/18 11 39 D NR  41  28 0 0
00114  2-32 81/04/18 11 27 B NN  84  28 0 0
00115  3-27 81/04/18 11 20 C NN  91  36 0 0
00116  3-32 81/04/18 11 07 B NN  71  25 0 0
00117  3-41 81/04/18 11 35 C NN  64  76 0 0
00118  2-24 81/04/18 11 22 A NR  48  28 0 0
00119  3-15 81/04/18 11 38 B NN  92  40 0 0
00120  3-29 81/04/18 11 14 A NN  72  33 0 0
00121  3-17 81/10/17 12 07 C E1  74  41 0 0
00122  2-07 81/10/17 12 18 B E1  78  86 0 0
00123  3-40 81/10/17 12 38 C E1  63  54 0 0
00124  2-22 81/10/17 12 32 C E1  67  58 0 0
00125  2-14 81/10/17 12 13 D E1  80  50 0 0
00126  1-33 81/10/17 12 22 C E1  52  40 0 0
00127  2-33 81/10/17 12 40 D E1  85  45 0 0
00128  3-31 81/10/17 12 17 B E1  79  41 0 0
00129  1-12 81/10/17 12 27 A E1  86  68 0 0
00130  1-46 81/10/17 12 02 C E1  54  50 0 0
00131  3-18 81/10/17 12 03 D E2  90  31 0 0
00132  2-38 81/10/17 12 23 D E2  55  44 0 0
00133  2-32 81/10/17 12 08 B E2  85  24 0 0
00134  3-13 81/10/17 12 28 D E2  59  27 0 0
00135  1-18 81/10/17 12 34 A E2  37  42 0 0
00136  1-35 81/10/17 12 06 D E2  82  58 0 0
00137  2-09 81/10/17 12 37 B E2  61  58 0 0
00138  1-30 81/10/17 12 12 A E2  50  38 0 0
00139  2-17 81/10/17 12 33 B E2  87  65 0 0
00140  3-29 81/10/17 12 19 A E2  70  23 0 0
00141  2-44 81/10/17 12 14 A U-  62  55 0 0
00142  3-07 81/10/17 12 25 A U-  78  34 0 0
00143  1-16 81/10/17 12 20 D U-  79  71 0 0
00144  1-31 81/10/17 12 39 D U-  67  44 0 0
00145  3-34 81/10/17 12 09 A U-  58  15 0 0
00146  2-41 81/10/17 12 31 B U-  75  31 0 0
00147  3-08 81/10/17 12 01 A U-  51  40 0 0
00148  1-14 81/10/17 12 05 B U-  61  57 0 0
00149  2-34 81/10/17 12 16 D U-  70  44 0 0
00150  3-38 81/10/17 12 29 B U-  81  20 0 0
```

Listing of the HIST File Data

```
00151  3-43 81/10/17 12 10 C NN  30  26 0 0
00152  1-17 81/10/17 12 04 A NN  92  22 0 0
00153  1-11 81/10/17 12 15 C NN  85  71 0 0
00154  3-21 81/10/17 12 36 A NN  87  42 0 0
00155  1-27 81/10/17 12 26 B NR  40  38 0 0
00156  1-25 81/10/17 12 21 B NN  75  25 0 0
00157  3-01 81/10/17 12 35 C NN  87  41 0 0
00158  2-20 81/10/17 12 30 D NN  58  47 0 0
00159  3-39 81/10/17 12 24 C NN  63  63 0 0
00160  3-20 81/10/17 12 11 C NR  80  41 0 0
00161  3-02 82/04/17 21 06 C E1  66  27 0 0
00162  2-21 82/04/17 21 13 A E1  81  40 0 0
00163  2-02 82/04/17 21 33 B E1  83  36 0 0
00164  2-26 82/04/17 21 18 D E1  48  25 0 0
00165  1-38 82/04/17 21 28 B E1  80  50 0 0
00166  1-13 82/04/17 21 01 A E1  53  51 0 0
00167  1-03 82/04/17 21 21 D E1  91  55 0 0
00168  2-24 82/04/17 21 12 A E1  50  36 0 0
00169  3-15 82/04/17 21 37 B E1  89  44 0 0
00170  2-01 82/04/17 21 40 C E1  77  51 0 0
00171  2-44 82/04/17 21 02 A E2  69  50 0 0
00172  2-07 82/04/17 21 29 B E2  72  27 0 0
00173  3-40 82/04/17 21 14 C E2  62  50 0 0
00174  1-11 82/04/17 21 24 C E2  87  79 0 0
00175  1-31 82/04/17 21 38 D E2  50  34 0 0
00176  3-01 82/04/17 21 07 C E2  90  43 0 0
00177  1-33 82/04/17 21 11 C E2  50  35 0 0
00178  2-34 82/04/17 21 17 D E2  71  47 0 0
00179  3-39 82/04/17 21 27 A E2  65  67 0 0
00180  3-20 82/04/17 21 34 C E2  77  38 0 0
00181  2-04 82/04/17 21 23 D U-  64  40 0 0
00182  2-05 82/04/17 21 08 A U-  71  42 0 0
00183  3-14 82/04/17 21 35 C U-  87  30 0 0
00184  1-24 82/04/17 21 03 D U-  58  22 0 0
00185  1-44 82/04/17 21 30 A U-  82  53 0 0
00186  3-03 82/04/17 21 05 D U-  44  23 0 0
00187  1-36 82/04/17 21 15 A U-  82  39 0 0
00188  1-40 82/04/17 21 10 B U-  63  46 0 0
00189  2-12 82/04/17 21 32 D U-  60  31 0 0
00190  2-35 82/04/17 21 20 A U-  87  52 0 0
00191  2-46 82/04/17 21 16 B NN  79  39 0 0
00192  2-25 82/04/17 21 39 A NR  58  58 0 0
00193  1-29 82/04/17 21 09 C NN  85  33 0 0
00194  2-37 82/04/17 21 25 C NN  79  57 0 0
00195  3-26 82/04/17 21 31 B NN  84  21 0 0
00196  1-39 82/04/17 21 19 B NN  81  62 0 0
00197  3-32 82/04/17 21 36 B NR  52  33 0 0
00198  2-45 82/04/17 21 04 D NN  51  44 0 0
00199  3-42 82/04/17 21 26 D NN  55  43 0 0
00200  1-51 82/04/17 21 22 B NN  86  59 0 0
```

Listing of the HIST File Data

00201	3-17	82/10/16	22	11	C	E1	69	40	0	0
00202	3-07	82/10/16	22	29	A	E1	88	45	0	0
00203	2-14	82/10/16	22	33	D	E1	79	57	0	0
00204	3-21	82/10/16	22	08	A	E1	86	20	0	0
00205	3-13	82/10/16	22	10	B	E1	60	35	0	0
00206	1-25	82/10/16	22	23	B	E1	69	24	0	0
00207	2-41	82/10/16	22	17	B	E1	79	39	0	0
00208	3-31	82/10/16	22	27	D	E1	80	34	0	0
00209	1-30	82/10/16	22	04	B	E1	48	34	0	0
00210	3-38	82/10/16	22	39	B	E1	83	25	0	0
00211	2-04	82/10/16	22	07	D	E2	68	39	0	0
00212	3-02	82/10/16	22	22	C	E2	76	26	0	0
00213	3-26	82/10/16	22	37	B	E2	89	29	0	0
00214	2-26	82/10/16	22	18	D	E2	49	23	0	0
00215	2-37	82/10/16	22	24	C	E2	78	50	0	0
00216	1-13	82/10/16	22	30	A	E2	58	66	0	0
00217	1-39	82/10/16	22	03	B	E2	83	59	0	0
00218	2-45	82/10/16	22	09	D	E2	56	43	0	0
00219	3-15	82/10/16	22	34	B	E2	87	40	0	0
00220	2-35	82/10/16	22	16	A	E2	89	42	0	0
00221	2-10	82/10/16	22	19	C	U-	69	67	0	0
00222	3-25	82/10/16	22	25	B	U-	73	6	0	0
00223	1-08	82/10/16	22	02	A	U-	82	61	0	0
00224	2-30	82/10/16	22	28	A	U-	38	54	0	0
00225	3-27	82/10/16	22	15	C	U-	84	48	0	0
00226	2-27	82/10/16	22	21	C	U-	59	35	0	0
00227	3-12	82/10/16	22	31	B	U-	87	44	0	0
00228	2-42	82/10/16	22	12	C	U-	75	49	0	0
00229	3-06	82/10/16	22	06	A	U-	92	36	0	0
00230	3-24	82/10/16	22	35	D	U-	70	21	0	0
00231	1-17	82/10/16	22	01	C	NR	93	36	0	0
00232	2-11	82/10/16	22	14	D	NR	87	62	0	0
00233	2-39	82/10/16	22	26	D	NN	61	40	0	0
00234	1-26	82/10/16	22	38	A	NN	52	47	0	0
00235	1-34	82/10/16	22	36	D	NN	75	18	0	0
00236	2-03	82/10/16	22	05	A	NN	69	66	0	0
00237	3-41	82/10/16	22	40	C	NR	55	70	0	0
00238	1-19	82/10/16	22	20	D	NN	44	46	0	0
00239	1-37	82/10/16	22	13	C	NN	88	33	0	0
00240	1-41	82/10/16	22	32	A	NN	79	62	0	0
00241	2-25	83/04/16	31	08	A	E1	55	50	0	0
00242	2-21	83/04/16	31	17	A	E1	82	38	0	0
00243	1-38	83/04/16	31	25	B	E1	82	45	0	0
00244	3-40	83/04/16	31	35	C	E1	65	48	0	0
00245	2-37	83/04/16	31	09	C	E1	76	49	0	0
00246	3-32	83/04/16	31	40	B	E1	50	34	0	0
00247	3-01	83/04/16	31	13	C	E1	88	39	0	0
00248	2-01	83/04/16	31	21	B	E1	81	49	0	0
00249	3-42	83/04/16	31	31	D	E1	62	46	0	0
00250	1-40	83/04/16	31	04	D	E1	70	48	0	0

Listing of the HIST File Data

```
00251  3-17 83/04/16 31 30 C E2  72  43 0 0
00252  2-10 83/04/16 31 05 A E2  72  65 0 0
00253  3-27 83/04/16 31 10 C E2  85  38 0 0
00254  2-30 83/04/16 31 14 A E2  42  50 0 0
00255  3-21 83/04/16 31 36 A E2  84  25 0 0
00256  3-12 83/04/16 31 18 B E2  86  49 0 0
00257  3-41 83/04/16 31 22 D E2  57  74 0 0
00258  3-06 83/04/16 31 03 A E2  90  38 0 0
00259  1-19 83/04/16 31 39 B E2  48  44 0 0
00260  2-42 83/04/16 31 26 C E2  71  48 0 0
00261  2-38 83/04/16 31 02 D U-  52  40 0 0
00262  3-23 83/04/16 31 29 A U-  90  60 0 0
00263  1-50 83/04/16 31 19 A U-  60  52 0 0
00264  1-43 83/04/16 31 06 C U-  80  60 0 0
00265  1-16 83/04/16 31 32 D U-  80  59 0 0
00266  2-09 83/04/16 31 27 B U-  61  67 0 0
00267  3-19 83/04/16 31 15 C U-  71  44 0 0
00268  1-28 83/04/16 31 24 A U-  50  45 0 0
00269  1-48 83/04/16 31 11 D U-  66  72 0 0
00270  2-36 83/04/16 31 37 B U-  87  48 0 0
00271  3-43 83/04/16 31 12 B NR  59  24 0 0
00272  3-04 83/04/16 31 34 D NR  78  32 0 0
00273  3-16 83/04/16 31 01 D NN  63  40 0 0
00274  1-21 83/04/16 31 23 B NN  41  17 0 0
00275  3-30 83/04/16 31 16 D NN  59  28 0 0
00276  1-45 83/04/16 31 33 C NN  89  61 0 0
00277  3-28 83/04/16 31 28 A NR  73  46 0 0
00278  1-09 83/04/16 31 07 D NN  74  59 0 0
00279  3-33 83/04/16 31 38 B NN  56  37 0 0
00280  3-39 83/04/16 31 20 C NR  62  63 0 0
```

APPENDIX B

THE ITEMBANK SYSTEM
DRAFT STAGE PROGRAMS

This appendix lists, in alphabetical order, the Draft Test Stage programs only. Figures 15 and 16 in the basic paper delineate the interrelationships of the programs. Programs for the steps in the remaining stages are very similar to those for the Draft Test Stage.

Readers are encouraged to test these programs on their own IBM PC using the data in the POOL and HIST files in the dummy database (see Appendix A). If possible, use a color monitor since instructions and messages are color coded. Remember that for these programs to operate properly, the following are required:

1. an IBM PC with a minimum of 128 K and two disk drives, a monitor, and a printer;

2. two data files, POOL and HIST, both on the B disk;

3. the ITEMBANK programs, all on the A disk;

4. dBASE II itself, on the A disk;

5. a file called TEST1, on the B disk. This file has the identical structure of the TEST2 file and, in fact, functions solely to provide the structure for the TEST2 file (see the createl program). The TEST1 file has only one record, consisting of all 1's.

Additional information on the programs is available from the author, John J. Bowers, Ph.D., The American College Testing Program, Health Programs Department, Area 29, Box 168, Iowa City, Iowa 52243, phone (319) 337-1063.

The contents of Appendix B are copyright © 1984 by The American College Testing Program, Inc. All rights reserved.

LIST OF PROGRAMS

Program Name	Page
altprint	B- 2
altlist.frm	B- 6
anal	B- 7
beeper	B- 8
bookin1	B- 9
bookin2	B-11
booknum	B-13
check1	B-15
create1	B-17
datain1	B-20
datain2	B-24
delete	B-26
draft	B-29
end	B-31
error	B-32
final	B-33
head1	B-34
head2	B-35
head3	B-36
init	B-37
itembank	B-39
jobend	B-41
kit	B-42
makealt	B-46
message1	B-48
message2	B-49
message3	B-50
newpage	B-51
print1	B-52
print2	B-53
print3	B-54
screen	B-55
sign	B-57
testinfo	B-58
testsum	B-59
tssprint	B-61
working	B-64

```
*  altprint.prg     01/30/84  jjb
*  called by draft.prg

*  All programs comprising the ITEMBANK system are
*  copyright (C) 1984 by The American College Testing Program, Inc.
*  All rights reserved.

*  uses the dBASE II report utility to print a simple list of the
*  alternate items.

*  set up the memory variables
restore from b:workm.mem additive
store '        Printing the Alternate Items List' to title

*  check to see that a draft test file exists.  if it doesn't, display a
*  warning message.  if it does, remind the user of the draft test form
*  number, administration date, and equator form numbers.

do while more1
     do head2
     if .not. file("b:test2")
        set color to 14,14
        @  7,10 say "No draft test file or alternate items list has " + ;
                    "been established."
        @  9,10 say "See your supervisor for further assistance."
        set color to 27,27
        @ 18,10 say "To return to the Draft Stage Menu, press M."
        @ 19,10 say "To quit, press any other key."
        set console off
        wait to answer
        set console on
        if !(answer) = "M"
           return
        else
           do end
        endif
     else
        do testinfo
        set color to 10,10
        @ 17,10 say "This step will print the alternate items list " + ;
                    "used in"
        @ 18,10 say "developing this test."
        set color to 27,27
        @ 20,10 say "To return to the Draft Stage Menu, press M."
        @ 21,10 say "To continue, press any other key."
        set console off
        wait to answer
        set console on
        if !(answer) = "M"
           return
        else
           store f to more1
        endif
     endif
enddo
```

```
*  allow the user to choose which type of items to print and in what
*  order they will be listed.  this may save time since the entire list
*  will not need to be reprinted if only a few changes are made to it.
do while more2
     do head2
     set color to 10,10
     @  7,10 say "Enter the number that represents the type of items " + ;
                 "you wish to print."
     @  8,10 say "Select only one."
     @ 10,15 say "1 = E1 items only."
     @ 12,15 say "2 = E2 items only."
     @ 14,15 say "3 = U- items only."
     @ 16,15 say "4 = NR items only."
     @ 18,15 say "5 = All items."
     set console off
     wait to answer
     set console on
     if &answer < 1 .or. &answer > 5
        do error
        loop
     endif
     do head2
     set color to 10,10
     @  7,10 say "Enter the letter that represents the order in " + ;
                 "which you"
     @  8,10 say "want the items listed.  Select only one."
     @ 10,15 say "A = By item ID number."
     if answer = '5'
        @ 12,15 say "B = By content classification, function, and "
        @ 13,15 say "    item ID number."
     else
        @ 12,15 say "B = By content classification and ID number."
     endif
     set console off
     wait to fx
     set console on
     store !(fx) to fx
     if fx <> "A" .and. fx <> "B"
        loop
     endif

*  set up the files needed to print the report
     do message2
     do case

        case answer <> '5'
        use b:pool
        copy to b:temp1 for f1 = '2' .and. f2 = answer
        use b:hist
        copy to b:temp2 for f3 = '2' .and. f4 = answer

        case answer = '5'
        use b:pool
        copy to b:temp1 for f1 = '2'
```

```
      use b:hist
      copy to b:temp2 for f3 = '2'

   endcase

   select primary
   use b:temp1
   select secondary
   use b:temp2
   select primary
   join to b:temp3 for p.id = s.id
   use b:temp3
   do case

      case fx = 'A'
      index on id to b:temp3x

      case fx = 'B'
      index on cont + func + id to b:temp3x

   endcase
   use b:temp3 index b:temp3x

*  make the graf field print the content field data in roman numerals.
   replace graf with '   I' for cont = '1'
   replace graf with '  II' for cont = '2'
   replace graf with ' III' for cont = '3'
   replace graf with '  IV' for cont = '4'

*  display a warning message to be sure the printer is on.
   do message1
   do message2
   do case

      case answer = '1'
      set heading to E1 Items Only

      case answer = '2'
      set heading to E2 Items Only

      case answer = '3'
      set heading to U- Items Only

      case answer = '4'
      set heading to NR Items Only

      case answer = '5'
      set heading to All Items

   endcase
   set console off
   eject
   report form altlist to print
   set console on
   use
```

```
      delete file b:temp1
      delete file b:temp2
      delete file b:temp3
      delete file b:temp3x.ndx
      store t to more3
      do while more3
         do head2
         set color to 10,10
         @  7,10 say "What do you want to do next?  Select only one."
         @  9,15 say "P = Print another Alternate Items List."
         @ 11,15 say "M = Return to the Draft Stage Menu."
         @ 13,15 say "Q = Quit."
         set console off
         wait to answer
         set console on
         do case

            case !(answer) = "P"
            store f to more3

            case !(answer) = "M"
            release all except st*
            return

            case !(answer) = "Q"
            do end

            otherwise
            do error
            loop

         endcase
      enddo
      loop
enddo

return
```

```
* altlist.frm -- called by altprint.prg
m=15, w=65
y
Alternate Items List
n
n
6,id + '  '
<Item¦ID #¦====
5,' ' + mkey + '    '
<Mas¦Key¦===
6,graf + '  '
<¦Cont¦====
5,fmt + '    '
<For¦mat¦===
6,' ' + form + '    '
<Test¦Form¦====
4,num
<¦Num¦===
5,diff
>¦Diff¦====
6,disc
>¦Disc¦====
```

```
*  anal.prg      02/02/84  jjb
*  called by itembank.prg

erase
set color to 12,12
@  1,19 say "========================================="
@  5,19 say "========================================="
set color to 11,11
@  3,19 say "TEST ANALYSIS AND ITEMBANK UPDATING STAGE"
set color to 10,10
@  5,80
text
After the test has been administered and the results analyzed, the new data

on the usage of each item must be entered to the database.  This involves

entering item performance data (difficulty and discrimination) and appending

the item data to the appropriate files:  the usage data to the HIST file and

the master data on new items to the POOL file.  Data used only during the te

development process are then deleted.  Once completed, the test development

cycle can begin again for the next test form.
endtext
set color to 14,14
@ 21,19 say "Programs for this stage are not available."
set color to 27,27
@ 23,19 say "Press any key to return to the Main Menu."
set console off
wait
set console on
return
```

```
*  beeper.prg      01/19/84  jjb
*  called by various programs

*  rings the bell three times as part of varous warning messages

store 0 to bell
do while bell <> 3
     @  0, 0 say chr(7)
     store bell + 1 to bell
enddo
release bell
return
```

```
*  bookin1.prg     01/29/84  jjb
*  called by booknum.prg

*  allows user to assign book numbers to items in the current draft test
*  file.  presents items in order by content classification, function,
*  and item ID number.

use b:test2 index b:test2x
do while .not. eof
     do head2
     store id to itemid
     store tbook to mtbook
     do screen
     set color to 142,142
     @ 13,48 say ">"
     set color to 27,27
     @ 19,57 say "Enter number + return. "
     @ 20,57 say "Press return alone to  "
     @ 21,57 say "leave number unchanged."
     @ 22,57 say "Enter 999 + return to  "
     @ 23,57 say "exit this step.        "
     set color to 29,10
     @ 13,49 get mtbook picture "###"
     read
     do case

        case mtbook = tbook
        store '   ' to mtbook
        store '    ' to itemid
        clear gets
        skip
        loop

        case mtbook = "999"
        store f to more2
        return

        otherwise
        set color to 29,29
        @ 13,49 say mtbook
        set color to 27,27
        @ 18,57 say "Press one + return:    "
        @ 19,57 say "V = verify entry and   "
        @ 20,57 say "    get next item;     "
        @ 21,57 say "X = verify entry and   "
        @ 22,57 say "    exit this step;    "
        @ 23,57 say "C = cancel and reenter."
        set console off
        wait to fx
        set console on
        do case

           case !(fx) = "V"
           replace tbook with mtbook
           store '   ' to mtbook
```

```
            store '     ' to itemid
            clear gets
            skip
            loop

            case !(fx) = "X"
            replace tbook with mtbook
            store f to more2
            return

            case !(fx) = "C"
            store '     ' to mtbook
            store '     ' to itemid
            clear gets
            loop

            otherwise
            do error
            store '     ' to mtbook
            store '     ' to itemid
            clear gets
            loop

        endcase
    endcase
enddo

return
```

```
*  bookin2.prg     01/28/84  jjb
*  called by booknum.prg

*  allows user to assign book numbers to items in the current draft test
*  file.  the user specifies the ID numbers of the items to receive book
*  numbers.

use b:test2 index b:test2x
do while more3
     do head2
     set color to 29,10
     @  7,10 say "Enter the ID number of the item you wish to see."
     @  9,10 say "Press return after your entry."
     @ 11,15 say "Item ID number:"
     @ 11,32 get itemid picture "#-##"
     read
     set color to 29,29
     @  11,32 say itemid
     set color to 27,27
     @ 13,10 say "Verify entry by pressing V."
     @ 14,10 say "Press any other key to cancel entry."
     @ 15,10 say "You may enter another number after cancelling " + ;
                 "this entry."
     set console off
     wait to fx
     set console on
     if !(fx) <> "V"
        store '    ' to itemid
        clear gets
        loop
     endif
     find &itemid
     if # = 0
        set color to 14,14
        @ 18,10 say "That item ID number is not in the draft test."
        @ 20,10 say "Check your data.  Press any key to continue."
        set console off
        wait
        set console on
        clear gets
        store '    ' to itemid
        loop
     endif
     store tbook to mtbook
     do head2
     do screen
     set color to 142,142
     @ 13,48 say ">"
     set color to 27,27
     @ 19,57 say "Enter number + return. "
     @ 20,57 say "Press return alone to  "
     @ 21,57 say "leave number unchanged."
     @ 22,57 say "Enter 999 + return to  "
     @ 23,57 say "exit this step.        "
     set color to 29,29
```

```
      @ 13,49 get mtbook picture "###"
      read
      do case

         case mtbook = tbook
         store '   ' to mtbook
         store '    ' to itemid
         clear gets
         loop

         case mtbook = "999"
         store f to more3
         return

         otherwise
         set color to 27,27
         @ 18,57 say "Press one + return:    "
         @ 19,57 say "V = verify entry and   "
         @ 20,57 say "    ask for next item; "
         @ 21,57 say "X = verify entry and   "
         @ 22,57 say "    exit this step;    "
         @ 23,57 say "C = cancel and reenter."
         set console off
         wait to fx
         set console on
         do case

            case !(fx) = "V"
            replace tbook with mtbook
            store '   ' to mtbook
            store '    ' to itemid
            clear gets
            loop

            case !(fx) = "X"
            replace tbook with mtbook
            store f to more3
            return

            case !(fx) = "C"
            store '   ' to mtbook
            store '    ' to itemid
            clear gets
            loop

            otherwise
            do error
            store '   ' to mtbook
            store '    ' to itemid
            clear gets
            loop

         endcase
      endcase
enddo
```

```
*  booknum.prg     01/28/84  jjb
*  called by draft.prg

*  assigns a control number called a "book number" to items selected for
*  the current draft test.  in this step, the user has two options with
*  respect to the order in which items appear.  in the first option, items
*  appear in order by content area, function, and item ID number.  this is
*  convenient for initial entry of book numbers since that is normally the
*  order in which book numbers are assigned.  the second option permits the
*  user to specify item ID's to view, a method more convenient if only a
*  few specific items need to have their book numbers assigned or changed.
*  in later stages, the user will have the option of displaying items by
*  content and item ID, book number, or test order number.

*  set up the memory variables
restore from b:workm.mem additive
store "              Assigning book numbers" to title

*  check to see that a draft test file exists.  if it doesn't, display
*  a warning message.  if it does, remind the user of the draft test
*  form number, administration date, and equator form numbers.
do while more1
     do head2
     if .not. file("b:test2")
        set color to 14,14
        @  7,15 say "No draft test file has been established."
        @  9,15 say "See your supervisor for further assistance."
        set color to 27,27
        @ 17,15 say "To return to the Draft Stage Menu, press M."
        @ 18,15 say "To quit, press any other key."
        set console off
        wait to answer
        set console on
        if !(answer) = "M"
           return
        else
           do end
        endif
     else
        do testinfo
        set color to 10,10
        @ 17,10 say "Book numbers entered in this step will be added to " +
                    "this draft "
        @ 18,10 say "test file."
        @ 19,10 say "To change an existing book number, simply enter " +
                    "the new one "
        @ 20,10 say "over the old one."
        set color to 27,27
        @ 22,10 say "To return to the Draft Stage Menu, press M."
        @ 23,10 say "To continue, press any other key."
        set console off
        wait to answer
        set console on
        if !(answer) = "M"
           return
```

```
         else
            store f to more1
         endif
      endif
enddo

*  allow the user to choose how items are to be displayed for data entry.
*  then enter book numbers according to this choice.
do while more2
      do head2
      set color to 10,10
      @  7, 5 say "How do you wish to have the items displayed?  " + ;
                  "Enter the letter"
      @  8, 5 say "of your choice."
      @ 10,10 say "A = In order by content classification, " + ;
                  "function, and ID number."
      @ 12,10 say "B = By specified item ID numbers.  A prompt will " + ;
                  "be displayed"
      @ 13,14 say "to permit entry of the desired ID numbers."
      @ 16, 5 say "Option A presents items in the order in which they " + ;
                  "normally receive"
      @ 17, 5 say "book numbers and is the more convenient option for " + ;
                  "initial entry of"
      @ 18, 5 say "the numbers.  It is also the more useful option for " + ;
                  "'browsing' through "
      @ 19, 5 say "the draft test file to review item data."
      @ 21, 5 say "Option B is more useful if you simply wish to " + ;
                  "change a few previously"
      @ 22, 5 say "entered book numbers."
      set console off
      wait to answer
      set console on
      do case

         case !(answer) = "A"
         use b:test2
         index on cont + f2 + id to b:test2x
         do bookin1

         case !(answer) = "B"
         use b:test2
         index on id to b:test2x
         do bookin2
         store f to more2

         otherwise
         do error
         loop

      endcase
enddo
release all except st*
delete file b:test2x.ndx
do message3
return
```

```
*  check1.prg     01/16/84  jjb
*  called by kit.prg

*  checks that form numbers entered are valid

   store '**********.**********************************************';
   to stars

*  check that the earlier form number is entered first and that
*  the same form number has not been entered twice for both equator forms.
   if meq1 >= meq2
      store 1 to check
      erase
      do beeper
      set color to 12,12
      @  0,10 say stars
      @  9,10 say stars
      set color to 13,13
      @  2,10 say "Your equator form numbers have not been entered " + ;
                  "correctly."
      @  4,10 say "Check to see that your form numbers are valid and have"
      @  5,10 say "been entered in the correct order."
      set color to 27,27
      @  7,10 say "Press any key to continue."
      set console off
      wait
      set console on
      return
   endif

*  check that equator form 1 exists
   select secondary
   goto top
   find &meq1
   if # = 0
      store 2 to check
      erase
      do beeper
      set color to 12,12
      @  0,10 say stars
      @  9,10 say stars
      set color to 13,13
      @  2,10 say "Equator form 1 cannot be found."
      @  4,10 say "Check your form numbers and reenter them."
      set color to 27,27
      @  7,10 say "Press any key to continue."
      set console off
      wait
      set console on
      return
      endif

*  check that equator form 2 exists
   select secondary
   goto top
```

```
find &meq2
if # = 0
   store 3 to check
   erase
   do beeper
   set color to 12,12
   @  0,10 say stars
   @  9,10 say stars
   set color to 13,13
   @  2,10 say "Equator form 2 cannot be found."
   @  4,10 say "Check your form numbers and reenter them."
   set color to 27,27
   @  7,10 say "Press any key to continue."
   set console off
   wait
   set console on
   return
endif

return
```

```
*  create1.prg     01/26/84  jjb
*  called by draft.prg

*  flags items in the pool and history files for use in a draft test
*  and creates a new database (b:test2) with data from these items for
*  use in future steps.  another program (tssprint) will print the
*  test summary sheet for this data.

*  set up the memory variables

restore from b:workm.mem additive
store "    Entering item ID's to create a draft test" to title

*  check to see if a draft test file already exists.  if it doesn't,
*  get information on the draft test to be created and create a data
*  file structure ready to receive data.  if it does, remind the user
*  of the draft test's form number, administration date, and equators.

do while more1
   do head2
   store '  ' to mtform
   store '        ' to mtdate
   store '  ' to meq1
   store '  ' to meq2
   set color to 29,10
   if .not. file("b:test2")
     @  7,10 say "Since there is no current draft test file, one must " + ;
                 "be established."
     @  8,10 say "Enter the following information in the order indicated."
     @  9,10 say "Enter blanks for all fields to return to the Draft " + ;
                 "Stage Menu."
     @ 10,10 say "Press return after each entry."
     @ 12,15 say "Draft test form number:"
     @ 14,15 say "Administration date (enter date"
     @ 15,15 say "numbers only in yy/mm/dd format):"
     @ 17,15 say "Number of equator form 1:"
     @ 19,15 say "Number of equator form 2:"
     clear gets
     @ 12,43 get mtform picture "##"
     @ 15,50 get mtdate picture "##/##/##"
     @ 17,43 get meq1 picture "##"
     @ 19,43 get meq2 picture "##"
     read
     if mtform = '  '
        return
     endif
     set color to 29,29
     @ 12,43 say mtform
     @ 15,50 say mtdate
     @ 17,43 say meq1
     @ 19,43 say meq2
     set color to 27,27
     @ 21,10 say "Verify entries by pressing V."
     @ 22,10 say "Cancel entries by pressing any other key."
     @ 23,10 say "You may then reenter the information correctly."
```

```
      set console off
      wait to answer
      set console on
      if !(answer) = "V"

*  check to see that the form numbers are valid and have been entered in
*  in the correct order
         select secondary
         use b:hist index b:histx2
         do check1
         if check <> 0
            clear gets
            loop
         endif

*  put the verified, checked draft test data in a memory file for use
*  in later steps.
         restore from b:testidm.mem additive
         store mtform to qm1
         store mtdate to qm2
         store meq1 to qm3
         store meq2 to qm4
         save to b:testidm.mem all like qm*
         release all like qm*

*  create the structure for the draft test file (b:test2)
         use b:test1
         copy to b:test2 structure
         store f to more1
      else
         clear gets
         loop
      endif
   else
      do testinfo
      set color to 10,10
      @ 17,10 say "Any item ID's entered in this step will be added to this"
      @ 18,10 say "draft test file.  To delete an item from the draft test,"
      @ 19,10 say "return to the Draft Stage Menu and select step D."
      set color to 27,27
      @ 22,10 say "To return to the Draft Stage Menu, press M."
      @ 23,10 say "To continue, press any other key."
      set console off
      wait to answer
      set console on
      if !(answer) = "M"
         return
      else
         store f to more1
      endif
   endif
enddo

store "B" to step
```

```
do datain1
do message2
do datain2
do message3
release all except st*
return
```

```
*  datain1.prg     01/27/84  jjb
*  called by create1.prg

*  adds item data from the b:pool file to the current draft test file or
*  alternate items list.  pool item flag fields (f1, f2) are changed
*  accordingly.  if the current step is "B", a record is appended to the
*  current draft test file and the pool data entered to this record.

select primary
use b:pool index b:poolx1
do case

     case step = "B"
     select secondary
     use b:test2

     case step = "C"
     select secondary
     use b:hist index b:histx1

endcase

do while more2
     do head2
     set color to 29,10
     clear gets
     store '     ' to itemid
     store '  ' to itemfunc
     @  7,10 say "For each item, enter the item's ID number and its " + ;
                 "function on the"
     @  8,10 say "examination:  E1, E2, U-, NR."
     @ 10,10 say "Press return after each entry."
     @ 12,10 say "Enter blanks for both fields to complete data entry."
     @ 14,15 say "Item ID number:"
     @ 16,15 say "Item function:"
     @ 14,32 get itemid picture "#-##"
     @ 16,32 get itemfunc picture "AX"
     read
     if itemfunc = '  '
        store f to more2
     else
        store !(itemfunc) to itemfunc
        set color to 29,29
        @ 14,32 say itemid
        @ 16,32 say itemfunc
        set color to 27,27
        @ 18,10 say "Verify entry by pressing V."
        @ 19,10 say "Cancel entry by pressing any other key."
        @ 20,10 say "You may continue to enter data after cancelling " + ;
                    "this entry."
        set console off
        wait to answer
        set console on
        if !(answer) = "V"
           select primary
```

```
            goto top
            find &itemid

*  if the item ID does not exist, warn the user
            if # = 0
               set color to 14,14
               @ 22,10 say "That item does not exist.  Check your data."
               @ 23,10 say "Press any key to continue."
               set console off
               wait
               set console on
               clear gets
               loop
            else

*  flag the item in the pool file (fields f1 and f2)
               set color to 14,14
               do case

               case step = "B"
               @ 22,10 say "Item data being entered to draft test file.  "+;
                           "Please wait."
               replace f1 with '1'

               case step = "C"
               @ 22,10 say "Item data being entered to alternate items " + ;
                           "list.  Please wait."
               replace f1 with '2'

               endcase

               do case

               case itemfunc = "E1"
               replace f2 with '1'

               case itemfunc = "E2"
               replace f2 with '2'

               case itemfunc = "U-"
               replace f2 with '3'

               case itemfunc = "NR"
               replace f2 with '4'

               otherwise
               do error
               clear gets
               loop

               endcase

*  if this is step B, append a record to the b:test2 file and copy
*  relevant pool item data into it.  if this is step C, change flag fields
*  f3 and f4 in the b:hist file.
```

```
do case

case step = "B"
select secondary
append blank
replace s.id with itemid, s.tfunc with itemfunc
replace s.tdate with mtdate, s.tform with mtform
replace s.mkey with p.mkey, s.cont with p.cont
replace s.fmt with p.fmt, s.lines with p.lines
replace s.lines with p.lines, s.graf with p.graf
replace s.f1 with p.f1, s.f2 with p.f2
clear gets
loop

case step = "C"
select secondary
goto top
store t to more3

   do case

   case itemfunc = "E1"
   find &itemid
   do while more3 .and. .not. eof
      if form = meq1
         replace f3 with '2', f4 with '1'
         store f to more3
      else
         skip
         loop
      endif
   enddo

   case itemfunc = "E2"
   find &itemid
   do while more3 .and. .not. eof
      if form = meq2
         replace f3 with '2', f4 with '2'
         store f to more3
      else
         skip
         loop
      endif
   enddo

   case itemfunc = "U-"
   find &itemid
   do while more3 .and. .not. eof
      skip
      if id <> itemid
         skip -1
         replace f3 with '2', f4 with '3'
         store f to more3
      else
         loop
```

```
                  endif
               enddo

               case itemfunc = "NR"
               find &itemid
               do while more3 .and. .not. eof
                  skip
                  if id <> itemid
                     skip -1
                     replace f3 with '2', f4 with '4'
                     store f to more3
                  else
                     loop
                  endif
               enddo

               endcase
            clear gets
            loop
            endcase

         endif
      else
         clear gets
         loop
      endif
   endif
enddo

return
```

```
*  datain2.prg     01/28/84  jjb
*  called by create1.prg

*  adds item data from the b:hist file to the current draft test file.
*  history item flag fields (f3, f4) are changed accordingly.  history
*  data are added to the record appended to the b:test2 file in the
*  datain1.prg just performed.

select primary
use b:hist index b:histx1
select secondary
use b:test2

*  change flag fields f3 and f4 in b:hist file.
select secondary
locate for f3 = ' '
do while .not. eof
     store id to itemid
     store tfunc to itemfunc
     store t to more3
     select primary
     goto top
     do case

        case itemfunc = "E1"
        replace f3 with '1', f4 with '1' for id = itemid .and. form = meq1

        case itemfunc = "E2"
        replace f3 with '1', f4 with '2' for id = itemid .and. form = meq2

        case itemfunc = "U-"
        find &itemid
        do while more3 .and. .not. eof
           skip
           if id <> itemid
              skip -1
              replace f3 with '1', f4 with '3'
              store f to more3
           else
              loop
           endif
        enddo

        case itemfunc = "NR"
        find &itemid
        do while more3 .and. .not. eof
           skip
           if id <> itemid
              skip -1
              replace f3 with '1', f4 with '4'
              store f to more3
           else
              loop
           endif
        enddo
```

```
        endcase

        goto top
        locate for p.id = itemid .and. p.f3 = '1'
        select secondary
        replace s.form with p.form, s.num with p.num, s.diff with p.diff
        replace s.disc with p.disc, s.f3 with p.f3, s.f4 with p.f4
        continue

enddo
```

```
*  delete.prg     01/26/84  jjb
*  called by draft.prg

*  deletes items from a draft test file or an alternate items list.
*  flag fields are changed accordingly in the b:pool and b:hist files.
*  in the draft test file (b:test2), deleted items are actually deleted as
*  contrasted with later steps in which deleted items are simply "flagged."

*  set up the memory variables
restore from b:workm.mem additive
if step = "D"
     store "         Deleting items from a draft test" to title
endif
if step = "E"
     store "    Deleting items from an alternate items list" to title
endif

*  check to see that a draft test file exists.  if it doesn't, display
*  a warning message.  if it does, remind the user of the draft test
*  form number, administration date, and equator form numbers.

do while more1
   do head2
   if .not. file("b:test2")
     set color to 14,14
     @  7,10 say "No draft test file or alternate items list has been " + ;
                 "established."
     @  9,10 say "See your supervisor for further assistance."
     set color to 27,27
     @ 18,10 say "To return to the Draft Stage Menu, press M."
     @ 19,10 say "To quit, press any other key."
     set console off
     wait to answer
     set console on
     if !(answer) = "M"
        return
     else
        do end
     endif
   else
     do testinfo
     set color to 10,10
     @ 17,10 say "Any item ID's entered in this step will be deleted " + ;
                 "from the"
     if step = "D"
        @ 18,10 say "current draft test file."
     endif
     if step = "E"
        @ 18,10 say "current alternate items list."
     endif
     set color to 27,27
     @ 21,10 say "To return to the Draft Stage Menu, press M."
     @ 22,10 say "To continue, press any other key."
     set console off
     wait to answer
```

```
      set console on
      if !(answer) = "M"
         return
      else
        store f to more1
      endif
    endif
enddo

*  set up the primary and secondary files
select primary
use b:pool index b:poolx1
select secondary
use b:hist index b:histx1

*  enable the user to enter the ID numbers of the items to be deleted.
*  change the flag fields in b:pool and b:hist accordingly.  if the item
*  is being deleted from the draft test, delete the item record from b:test
do while more2
     do head2
     set color to 29,10
     clear gets
     store '    ' to itemid
     @  7,10 say "Enter the ID number of the item that you wish to delete."
     @  9,10 say "Press return after each entry."
     @ 11,10 say "Enter blanks to complete data entry."
     @ 13,15 say "Item ID number:"
     @ 13,32 get itemid picture "#-##"
     read
     if itemid = ' -  '
        store f to more2
     else
        set color to 29,29
        @ 13,32 say itemid
        set color to 27,27
        @ 18,10 say "Verify entry by pressing V."
        @ 19,10 say "Cancel entry by pressing any other key."
        @ 20,10 say "You may continue to enter ID numbers after " + ;
                    "cancelling this entry."
        set console off
        wait to answer
        set console on
        if !(answer) = "V"
           set color to 14,14
           select primary
           goto top
           find &itemid
           if # = 0
              @ 22,10 say "That item ID does not exist.  Check your data."
              @ 23,10 say "Press any key to continue."
              set console off
              wait
              set console on
              loop
           else
```

```
            @ 22,10 say "Item data now being deleted.  Please wait."
            replace f1 with '0', f2 with '0'
         endif
         select secondary
         goto top
         find &itemid
         do while id = itemid .and. .not. eof
            if f3 <> '0'
               replace f3 with '0', f4 with '0'
            else
               skip
               loop
            endif
         enddo
         if step = "D"
            use b:test2
            delete for id = itemid
            pack
            select primary
            use b:pool index b:poolx1
         endif
         clear gets
         loop
      else
         loop
      endif
   endif
enddo

do message3
release all except st*
return
```

```
*  draft.prg     01/15/84  jjb
*  called by itembank.prg

*  display menu for draft stage and perform step selected by the user..

*  set up the loop
   store t to start
   do while start

   *  set up screen and prompts
      erase
      set color to 12,12
      @  0,23 say "=================================="
      @  4,23 say "=================================="
      set color to 11,11
      @  2,23 say "DRAFT STAGE MENU -- ITEMBANK SYSTEM"
      set color to 10,10
      @  6, 5 say "Enter the letter for the step you wish to perform " + ;
                  "or Q to quit."
      @  8, 5 say "A = Print a Test Drafting Kit."
      @  9, 5 say "B = Enter item ID's to create a draft test, OR " + ;
                  "add item ID's to the"
      @ 10, 5 say "    current draft test."
      @ 11, 5 say "C = Enter item ID's to create an alternate items " + ;
                  "list OR add item ID's"
      @ 12, 5 say "    to the current alternate items list."
      @ 13, 5 say "D = Delete items from the draft test."
      @ 14, 5 say "E = Delete items from the alternate items list."
      @ 15, 5 say "F = Assign book numbers to the draft test items OR " + ;
                  "browse through the "
      @ 16, 5 say "    draft test items."
      @ 17, 5 say "G = Print a Test Summary Sheet for the draft test."
      @ 18, 5 say "H = Print the alternate items list."
      @ 20, 5 say "M = Return to the Main Menu."
      @ 22, 5 say "Q = Quit."

      set console off
      wait to step
      set console on
      store !(step) to step

   *  select program for appropriate step
      do case

         case step = "A"
         do kit

         case step = "B"
         do create1

         case step = "C"
         do makealt

         case step = "D"
         do delete
```

```
            case step = "E"
            do delete

            case step = "F"
            do booknum

            case step = "G"
            do tssprint

            case step = "H"
            do altprint

            case step = "M"
            store f to start

            case step = "Q"
            do end

            otherwise
            do error
            loop

         endcase

*     loop back again
      enddo

      return
```

```
*  end.prg      01/15/84  jjb
*  called by various programs

*  clear all current files
clear

*  check to see if a backup of data file is needed

do while t
erase
set color to 12,12
@  1,10 say "**********************************************************************
@  9,10 say "**********************************************************************
set color to 14,14
@  3,10 say "Have you changed any records in the database?"
@  5,10 say "If yes, use diskcopy to make a new backup of the changed files.
@  7,10 say "For help, see your microcomputer consultant."
set color to 27,27
@ 15,10 say "Press Q to quit to the operating system."
@ 17,10 say "Press D to return to dBASE II."

set console off
wait to end
set console on

do case

   case !(end) = 'Q'
   release all
   use
   quit

   case !(end) = 'D'
   set talk on
   set bell on
   set eject on
   set confirm on
   release all
   use
   erase
   set color to 14,14
   @  0, 0 say "The message DO CANCELLED below indicates that you have " + ;
               "returned to dBASE II."
   set color to 112,07
   cancel

   otherwise
   loop

endcase

enddo
```

```
*  error.prg     01/15/84  jjb
*  called by itembank.prg and others

*  erase menu and display error message

erase

do beeper

set color to 12,12
@  3,17 say "****************************************"
@ 15,17 say "****************************************"
set color to 13,13
@  8,25 say "Invalid option selected."
@ 10,32 say "Try again."
@ 12,24 say "Press any key to continue."
set color to 140,140
@  6,34 say "ERROR"

set console off
wait
set console on

set color to 112,7

return
```

```
*  final.prg      02/02/84  jjb
*  called by itembank.prg

erase
set color to 12,12
@  1,23 say "=================================="
@  5,23 say "=================================="
set color to 11,11
@  3,23 say "FINAL STAGE MENU -- ITEMBANK SYSTEM"
@  5,80
set color to 10,10
text
Once comments are received from reviewes, the working test becomes a final

test.  Recommended changes, possibly including further substitution of alter

nate items, key changes, etc., are incorporated.  The step ends when the

revised test is printed and administered.
endtext
set color to 14,14
@ 21,19 say "Programs for this stage are not available."
set color to 27,27
@ 23,19 say "Press any key to return to the Main Menu."
set console off
wait
set console on
return
```

```
*  head1.prg     01/17/84  jjb
*  called by kit.prg

*  prints heading for the four parts of the test drafting kit

@  3,14 say topline
@  3,48 say str(ctr1,1)
@  4,12 say 'Items available for selection as'
@  4,45 say itemfunc
@  4,48 say 'items'
@  5,24 say "printed " + date()
@  7, 7 say 'Mas'
@  7,13 say 'For'
@  7,19 say 'Con'
@  7,40 say 'Form   Func'
@  8, 0 say 'ID #   Key   mat   tent'
@  8,27 say 'Form   Num   Key    tion   Diff   Disc'
@  9, 0 say '----   ---   ---   ----'
@  9,27 say '----   ---   ----   ----   ----   ----'
store 10 to ctr2
store 1 to pagenum
return
```

```
*  head2.prg     01/21/84  jjb
*  called by various programs

*  shows a heading on the screen for the step being performed

erase
set color to 9,9
@  1,15 say "================================================"
@  5,15 say "================================================"
set color to 11,11
@  3,15 say title
return
```

```
*  head3.prg      01/30/84  jjb
*  called by tssprint.prg

*  prints the heading for the Test Summary Sheet in the draft test
*  stage.  similar programs print slightly different headings in later
*  stages.  for example, later stages and steps will include columns
*  for line length and graphics, if needed.  the final version of the
*  Test Summary Sheet, printed after the test is administered, will
*  include item difficulties and discriminations from that administration.

set margin to 5
@  3,26 say topline
@  4,21 say "Form "+ mtform + "/Administration Date " + mtdate
@  5,31 say "printed " + date()
@  8, 3 say "Test Development Data"
@  8,34 say "Master Item Data"
@  8,58 say "Latest Usage Data"
@  9, 0 say "----------------------------"
@  9,32 say "--------------------"
@  9,56 say "--------------------"
@ 10, 0 say "Book  Repl  Test  Form  Func"
@ 10,32 say "Item  Mas  Con-  For"
@ 11, 0 say "Num   By    Seq#  Key   tion"
@ 11,32 say "ID #  Key  tent  mat"
@ 11,56 say "Form  Num  Diff  Disc"
@ 12, 0 say "====  ====  ====  ====  ====="
@ 12,32 say "====  ===  ====  ==="
@ 12,56 say "====  ===  ====  ===="
store 1 to pagenum
store 13 to ctr2
```

```
*  init.prg     01/15/84  jjb
*  called by itembank.prg
*  place here all commands that affect the entire system

*  define environment
   set talk off
   set bell off
   set eject off
   set confirm on
   set colon off

*  set up copyright notice in memory variables.  thus, anytime that the
*  itembank program is in operation, these words will be in memory.

   store "This program is protected by U.S. copyright law"       to copymem1
   store "(Title 17 United States Code).  Unauthorized"          to copymem2
   store "reproduction and/or sales may result in imprisonment"  to copymem3
   store "of up to 1 year and fines up to $10,000 (17 USC 506)." to copymem4
   store "Copyright infringers may also be subject to civil"     to copymem5
   store "liability."                                            to copymem6
   store "Copyright (C) 1984 by The American College Testing"    to copymem7
   store "Program, Inc.  All rights reserved."                   to copymem8
   save to copyrite.mem
   release all

*  check for b:poolm.mem; if not there, initialize memory variables,
*  save, and clear memory

   if .not. file("b:poolm.mem")
      store '     ' to mid
      store ' '     to mmkey
      store ' '     to mcont
      store '  '    to mfmt
      store '  '    to mlines
      store '  '    to mstatus
      store '     ' to mgraf
      store ' '     to mf1
      store ' '     to mf2
      save to b:poolm
      release all
   endif

*  check for b:histm.mem; if not there, initialize memory variables,
*  save, and clear memory

   if .not. file("b:histm.mem")
      store '     '      to mid
      store '         ' to mdate
      store '  '         to mform
      store '  '         to mnum
      store ' '          to mfkey
      store '  '         to mfunc
      store 00           to mdiff
      store 00           to mdisc
      store ' '          to mf3
```

```
      store ' '          to mf4
      save to b:histm
      release all
   endif

*  check for b:workm.mem; if not there, initialize memory variables,
*  save, and clear memory

   if .not. file ("b:workm.mem")
      store t            to more1
      store t            to more2
      store t            to more3
      store 0            to check
      store 00           to ctr1
      store 00           to ctr2
      store 00           to pagenum
      store ' '          to fx
      store ' '          to answer
      store '  '         to mtform
      store '  '         to meq1
      store '  '         to meq2
      store '  '         to itemfunc
      store '   '        to mtbook
      store '    '       to itemid
      store '         ' to mtdate
      store '                                                    ' to title
      store '                                                ' to topline
      save to b:workm
      release all
   endif

*  check for b:testidm.mem.; if not there, initialize memory variables,
*  save, and clear memory
   if .not. file("b:testidm.mem")
      store '  '         to qm1
      store '         ' to qm2
      store '  '         to qm3
      store '  '         to qm4
      save to b:testidm
      release all
   endif

*  set up files and indexes

*  use b:pool
*  index on id to b:poolx1
*  index on cont + id to b:poolx2
*  use b:hist
*  index on id + form to b:histx1
*  index on form + id to b:histx2

   return
```

```
*  itembank.prg     01/15/84  jjb
*  main program of the itembank database management system

*  copyright (c) 1984 by
*  The American College Testing Program, Inc.
*  all rights reserved

*  display signon message while initializing
   do sign

*  initialize variables, set up environment, use files, etc.
   do init

*  set up the loop
   do while t

      *  set up screen and prompts
         erase
         set color to 12,12
         @  1,26 say "============================"
         @  5,26 say "============================"
         set color to 14,14
         @  3,26 say "MAIN MENU -- ITEMBANK SYSTEM"
         set color to 10,10
         @  7, 5 say "Enter the number for your current stage of test " + ;
                     "development or 5 to quit."
         @  9,10 say "1 = Draft test stage."
         @ 11,10 say "2 = Working test stage."
         @ 13,10 say "3 = Final test stage."
         @ 15,10 say "4 = Test analysis and itembank updating stage."
         @ 17,10 say "5 = Quit."
         set console off
         wait to stage
         set console on

      *  select test development stage
         do case

            case stage = '1'
            do draft

            case stage = '2'
            do working

            case stage = '3'
            do final

            case stage = '4'
            do anal

            case stage = '5'
            do end

            otherwise
            loop
```

```
      endcase

*     loop back again
      enddo
```

```
*  jobend.prg     01/18/84  jjb
*  called by various programs

*  informs the user that a particular job has been completed and permits
*  user to quit or continue

do while t
erase
set color to 13,13
@  5,10 say "**********************************************************"
@ 15,10 say "**********************************************************"
set color to 14,14
@  7,10 say "Job completed."
@  9,10 say "Do you wish to quit now?"
set color to 27,27
@ 11,10 say "If yes, press Y."
@ 13,10 say "If no, press N to return to the menu for this stage."
set console off
wait to answer
set console on
do case

     case !(answer) = 'Y'
     do end

     case !(answer) = 'N'
     return

     otherwise
     do error
     loop

endcase
enddo
```

```
*  kit.prg     01/17/84  jjb
*  called by draft.prg

*  prints all data for items eligible for selection for use in the draft
*  test currently being developed

*  set up the memory variables
   restore from b:workm.mem additive
   store 'ITEMBANK Test Drafting Kit - Part' to topline
   store '            Printing a Test Drafting Kit' to title

*  set up the loop
   do while more1
      do head2

*  check to see if a draft test file exists.  if it does, display a
*  warning message.
   if file("b:test2")
      set color to 140,140
      @  7,35 say "WARNING"
      do beeper
      set color to 14,14
      @  9,15 say "A draft test file currently exists."
      @ 11,15 say "This step should not be performed if a "
      @ 12,15 say "draft test file exists."
      set color to 27,27
      @ 20,15 say "To return to the Draft Stage Menu, press M."
      @ 21,15 say "To quit, press any other key."
      set console off
      wait to answer
      set console on
      if !(answer) = "M"
         return
      else
         do end
      endif
   endif

*  enter the form numbers of the equator test forms
      set color to 29,10
      @  7,10 say "Enter the numbers of the test forms to which " + ;
                  "this test form"
      @  8,10 say "is to be equated."
      @ 10,10 say "Enter blanks to return to the Draft Stage Menu."
      @ 12,10 say "Be sure to press the return key after each entry."
      @ 14,15 say "Equator form 1:  "
      @ 16,15 say "Equator form 2:  "
      @ 14,32 get meq1 picture "##"
      @ 16,32 get meq2 picture "##"
      read
      if meq1 = '  ' .or. meq2 = '  '
         return
      endif
      set color to 29,29
      @ 14,32 say meq1
```

```
      @ 16,32 say meq2
      set color to 27,27
      @ 20,10 say "Verify entries by pressing V."
      @ 21,10 say "Cancel entries by pressing any other key."
      @ 22,10 say "You may then reenter the information correctly."
      set console off
      wait to answer
      set console on
      if !(answer) <> "V"
         clear gets
         store '  ' to meq1
         store '  ' to meq2
         loop
      endif

*  set up primary and secondary files
      select primary
      use b:pool index b:poolx1
      select secondary
      use b:hist index b:histx2

*  check to see that form numbers are valid.
      do check1
      if check > 0
         clear gets
         loop
      endif

*  show a screen to indicate files are being processed.
      do message2

*  set up codes in primary file to identify equator, useo,
*  and new-revised items.

      select secondary
      goto top
      find &meq1
      do while form = meq1 .and. .not. eof
         store id to itemid
         select primary
         find &itemid
         if status = "AC"
            replace f1 with '1'
         endif
         select secondary
         skip
      enddo

      select secondary
      goto top
      find &meq2
      do while form = meq2 .and. .not. eof
         store id to itemid
         select primary
         find &itemid
```

```
        if status = "AC" .and. f1 <> '1'
           replace f1 with '2'
        endif
        select secondary
        skip
      enddo

     select primary
     replace f1 with '3' for status = 'AC' .and. f1 = '0'
     replace f1 with '4' for status = 'AR'

*  change the primary and secondary files.
     select primary
     use b:pool index b:poolx2
     select secondary
     use b:hist index b:histx1

*  show warning message to turn printer on.
     do message1

*  show (again) a screen to indicate records are being processed.
     do message2

*  print out item listings for each item eligibility category:
*  E1, E2, U- and NR.  the subroutine below makes four passes through
*  the files, printing data on E1, E2, U- and NR items in succession.
     set margin to 9
     store ctr1 + 1 to ctr1
     set format to print
     eject
     do while ctr1 <= 4

        do case

           case ctr1 = 1
           store "E1" to itemfunc

           case ctr1 = 2
           store "E2" to itemfunc

           case ctr1 = 3
           store "U-" to itemfunc

           case ctr1 = 4
           store "NR" to itemfunc

        endcase

        do head1

        select primary
        goto top
        do while .not. eof
          if val(f1) = ctr1
             store id to itemid
```

```
            if ctr2 >= 60
               do newpage
            endif
            do print1
            select secondary
            goto top
            find &itemid
            do while id = itemid .and. .not. eof
              do print2
              skip
            enddo
            select primary
          endif
            skip
        enddo

        set print on
        eject
        set print off

        store ctr1 + 1 to ctr1

    enddo

    use b:pool
    replace all f1 with '0'

    set format to screen
    do jobend
    store f to more1

enddo

release all except st*

return
```

```
*  makealt.prg     01/27/84  jjb
*  called by draft.prg

*  flags items in the pool and history files for use as alternate items.
*  in a later stage, such items may be substituted for items in the
*  draft test.  unlike the draft test items, alternate items are not
*  placed in a separate file since they are not handled as extensively.
*  another program (altprint) will print a summary sheet on these items.

*  set up the memory variables
restore from b:workm.mem additive
store "   Entering item ID's to an alternate items list" to title

*  check to see that a draft test file exists.  if it doesn't, display
*  a warning message.  if it does, remind the user of the draft test
*  form number, administration date, and equator form numbers.
do while more1
     do head2
     if .not. file("b:test2")
        set color to 10,10
        @  7,10 say "No draft test file has been established."
        @  9,10 say "An alternate items list should not be created before"
        @ 10,10 say "the draft test file has been created."
        set color to 27,27
        @ 20,10 say "To return to the Draft Stage Menu, press M."
        @ 21,10 say "To quit, press any other key."
        set console off
        wait to answer
        set console on
        if !(answer) = "M"
           return
        else
          do end
        endif
     else
        do testinfo
        set color to 10,10
        @ 17, 5 say "Any item ID's entered in this step will be added to "+;
                    "the alternate items"
        @ 18, 5 say "list for use in developing this test."
        @ 19, 5 say "To delete an item from the list, return to the Draft "+;
                    "Stage Menu and "
        @ 20, 5 say "select step E."
        set color to 27,27
        @ 22,10 say "To return to the Draft Stage Menu, press M."
        @ 23,10 say "To continue, press any other key."
        set console off
        wait to answer
        set console on
        if !(answer) = "M"
           return
        else
           store f to more1
        endif
     endif
```

```
enddo
do datain1
do message3
release all except st*
return
```

```
*  message1.prg     01/17/84  jjb
*  called by various programs

*  set up warning message for printer

set format to screen
erase
do beeper
set color to 13,13
@  2,15 say "****************************************"
@ 13,15 say "****************************************"
set color to 9,9
@  4,20 say "The job is now ready to print."
@  6,20 say "Be sure printer is turned "
@  8,20 say "and on-line indicator is      ."
@ 10,20 say "Press any key to continue."
set color to 168,168
@  6,46 say " ON "
@  8,45 say " LIT "
set console off
wait
set console on
set color to 112,07
return
```

```
*  message2.prg     01/16/84  jjb
*  called by various programs

*  show a message to indicate files are being processed.

erase
set color to 9,9
@  5,20 say "******************************"
@ 11,20 say "******************************"
set color to 10,10
@  7,20 say "Files are now being processed."
@  9,20 say "Please wait."
set color to 112,07
return
```

```
*  message3.prg     01/21/84  jjb
*  called by various programs

*  displays a message indicating that files have been processed and
*  that a hard copy listing of the test summary sheet can be obtained
*  using the tssprint.prg.

do while t
erase
set color to 13,13
@  1,10 say "*************************************************************"
@ 19,10 say "*************************************************************"
set color to 14,14
@  5,10 say "If you want a printed copy of the current version of the "
@  7,10 say "Summary Sheet for either the draft test or the alternate"
@  9,10 say "items list, return to the menu for this stage and select"
@ 11,10 say "the appropriate option."
set color to 27,27
@ 15,10 say "If yes, press Y."
@ 17,10 say "If no, press N to return to the menu for this stage."
set color to 11,11
@  3,31 say "Job completed."
@ 13,25 say "Do you wish to quit now?"
set console off
wait to answer
set console on
do case

     case !(answer) = 'Y'
     do end

     case !(answer) = 'N'
     return

     otherwise
     do error
     loop

endcase

enddo
```

```
*  newpage.prg     01/18/84  jjb
*  called from various programs

*  when the end of a page is reached (i.e., line counter >= 60),
*  this program ejects the page, prints a page heading, and resets the
*  line counter.

store pagenum + 1 to pagenum
@  3, 0 say topline
if step = "A"
     @  3,34 say str(ctr1,1)
endif
@  3,59 say 'page'
@  3,64 say str(pagenum,1)
@  4, 0 say "printed " + date()
store 5 to ctr2
return
```

```
*  print1.prg     01/17/84 jjb
*  called by kit.prg

*  prints pool data for each item

store ctr2 + 1 to ctr2
@ ctr2, 0 say itemid
@ ctr2, 8 say mkey
@ ctr2,14 say fmt
do case

     case cont = '1'
     @ ctr2,20 say "  I"

     case cont = '2'
     @ ctr2,20 say " II"

     case cont = '3'
     @ ctr2,20 say "III"

     case cont = '4'
     @ ctr2,20 say " IV"

endcase
return
```

```
*  print2.prg     01/17/84  jjb
*  called by kit.prg

*  prints history data for each item in the test drafting kit

@ ctr2,28 say form
@ ctr2,35 say num
@ ctr2,41 say fkey
@ ctr2,48 say func
@ ctr2,55 say diff
@ ctr2,62 say disc

store ctr2 + 1 to ctr2
return
```

```
*  print3.prg     01/31/84  jjb
*  called by tssprint.prg and other programs

*  prints each line of the item data for the Test Summary Sheet.
*  in the Draft Test Stage, some of these columns will be blank.

set margin to 5
store ctr2 + 1 to ctr2
@ ctr2,  1 say tbook
@ ctr2,  7 say trepl
@ ctr2,13 say tnum
@ ctr2,20 say tfkey
@ ctr2,25 say tfunc
@ ctr2,32 say id
@ ctr2,39 say mkey
do case

     case cont = '1'
     @ ctr2,44 say "  I"

     case cont = '2'
     @ ctr2,44 say " II"

     case cont = '3'
     @ ctr2,44 say "III"

     case cont = '4'
     @ ctr2,44 say " IV"

endcase
@ ctr2,50 say fmt
@ ctr2,57 say form
@ ctr2,63 say num
@ ctr2,68 say diff
@ ctr2,74 say disc

return
```

```
*  screen.prg     01/28/84  jjb
*  called by various programs

*  this program is essentially a "shell" for displaying and changing
*  data on items in the current draft test file.  a flashing yellow ">"
*  marks the fields in which data may be entered or in which existing data
*  may be changed.  this is a basic screen used for data entry or updating
*  throughout the itembank system.  the "get" statements in the calling
*  program allow selected fields to receive data while locking out and
*  protecting others from any change.

set color to 10,10
@  7,19 say "*** Information on item ID number      ***"
set color to 9,9
@ 10, 0 say "--------------------------"
@ 10,33 say "--------------------"
@ 10,59 say "--------------------"
set color to 10,10
@  9, 0 say "     Master item data"
@  9,33 say " Current test data"
@  9,59 say " Latest usage data"
@ 11, 0 say "Content class"
@ 13, 0 say "Format"
@ 15, 0 say "Master key"
@ 17, 0 say "Length (#lines)"
@ 19, 0 say "Graphic (type/#lines)"
@ 11,33 say "Test order #"
@ 13,33 say "Book number"
@ 15,33 say "Replacement"
@ 17,33 say "Form key"
@ 19,33 say "Function"
@ 21,33 say "Difficulty"
@ 23,33 say "Discrimination"
@ 11,59 say "Form"
@ 13,59 say "Number"
@ 15,59 say "Difficulty"
@ 17,59 say "Discrimination"
set color to 29,29
@  7,53 say itemid
do case

     case cont = '1'
     @ 11,22 say "  I"

     case cont = '2'
     @ 11,22 say " II"

     case cont = '3'
     @ 11,22 say "III"

     case cont = '4'
     @ 11,22 say " IV"

endcase
@ 13,22 say fmt
```

```
@ 15,22 say mkey
@ 17,22 say lines
@ 19,22 say graf
@ 11,49 say tnum
@ 13,49 say tbook
@ 15,49 say trepl
@ 17,49 say tfkey
@ 19,49 say tfunc
@ 21,49 say tdiff
@ 23,49 say tdisc
@ 11,76 say form
@ 13,76 say num
@ 15,76 say diff
@ 17,76 say disc
```

```
*  sign.prg     01/15/84  jjb
*  called by itembank.prg

*  displays a sign-on message while program is initializing environment.

erase
set color to 14,14
@  2,17 say "Welcome to the demonstration version of"
set color to 141,141
@  4,32 say "ITEMBANK"
set color to 10,10
@  6,20 say "A database management system for"
@  8,20 say "item banking and test development"
@ 10,31 say "Version 1.1"
set color to 12,12
@ 12,30 say "**************"
set color to 9,9
@ 14,26 say "John J. Bowers, Ph.D."
@ 15,28 say "Program Associate"
@ 16,23 say "Health Programs Department"
@ 18,15 say "The American College Testing Program, Inc."
set color to 13,13
@ 20, 5 say "Copyright (C) 1984 by The American College Testing Program, Inc
@ 21,27 say "All rights reserved."
set color to 27,27
@ 24, 7 say "The system environment is being initialized.  Please wait."
set color to 112,07
return
```

```
*  testinfo.prg     01/28/84  jjb
*  called by various programs

*  displays basic information regarding the current test form under
*  development:  form number, administration date, and equator form
*  numbers.  used in various programs as a sign-on reminder to the user.

restore from b:testidm.mem additive
set color to 10,10
@  7,10 say "A draft test file has been established for the " + ;
            "following test:"
@  9,15 say "Draft test form number:"
@ 11,15 say "Administration date:"
@ 13,15 say "Number of equator form 1:"
@ 15,15 say "Number of equator form 2:"
set color to 29,29
@  9,43 say qm1
@ 11,37 say qm2
@ 13,43 say qm3
@ 15,43 say qm4
store qm1 to mtform
store qm2 to mtdate
store qm3 to meq1
store qm4 to meq2
release all like qm*
use b:test2
return
```

```
*  testsum.prg     01/31/84  jjb
*  called by tssprint.prg and other programs

*  computes and prints a summary of item statistics for the test
*  currently under development.

use b:test2
copy to b:temp field id, cont, f1, f2, diff, disc for f1 = '1'
use b:temp
count for f1 = '1' to cntot
count for f1 = '1' .and. f2 <> '4' to cntota
count for f2 = '1' to cne1
count for f2 = '2' to cne2
count for f2 = '3' to cnuu
count for f2 = '4' to cnnr
count for cont = '1' to cni
count for cont = '2' to cnii
count for cont = '3' to cniii
count for cont = '4' to cniv
count for cont = '1' .and. f2 <> '4' to cnia
count for cont = '2' .and. f2 <> '4' to cniia
count for cont = '3' .and. f2 <> '4' to cniiia
count for cont = '4' .and. f2 <> '4' to cniva
sum diff to cftot for f2 <> '4'
sum diff to cfe1 for f2 = '1'
sum diff to cfe2 for f2 = '2'
sum diff to cfuu for f2 = '3'
sum diff to cfi for cont = '1' .and. f2 <> '4'
sum diff to cfii for cont = '2' .and. f2 <> '4'
sum diff to cfiii for cont = '3' .and. f2 <> '4'
sum diff to cfiv for cont = '4' .and. f2 <> '4'
sum disc to cstot for f2 <> '4'
sum disc to cse1 for f2 = '1'
sum disc to cse2 for f2 = '2'
sum disc to csuu for f2 = '3'
sum disc to csi for cont = '1' .and. f2 <> '4'
sum disc to csii for cont = '2' .and. f2 <> '4'
sum disc to csiii for cont = '3' .and. f2 <> '4'
sum disc to csiv for cont = '4' .and. f2 <> '4'
set margin to 17
@  8, 8 say "Summary of Item Statistics"
@  9, 0 say "---------------------------------------"
@ 10,21 say "# of    Mean   Mean "
@ 11, 0 say "Breakdown by"
@ 11,21 say "Items   Diff   Disc"
@ 12, 0 say "=================="
@ 12,21 say "=====   ====   ====="
@ 14, 0 say "Item Function   E1"
@ 14,22 say str(cne1,2)
@ 14,30 say str(cfe1/cne1,2)
@ 14,37 say str(cse1/cne1,2)
@ 15,16 say "E2"
@ 15,22 say str(cne2,2)
@ 15,30 say str(cfe2/cne2,2)
@ 15,37 say str(cfe2/cne2,2)
```

```
@ 16,16 say "U-"
@ 16,22 say str(cnuu,2)
@ 16,30 say str(cfuu/cnuu,2)
@ 16,37 say str(csuu/cnuu,2)
@ 17,16 say "NR"
@ 17,22 say str(cnnr,2)
@ 17,30 say "--"
@ 17,37 say "--"
@ 19, 0 say "Content Class    I"
@ 19,22 say str(cni,2)
@ 19,30 say str(cfi/cnia,2)
@ 19,37 say str(csi/cnia,2)
@ 20,16 say "II"
@ 20,22 say str(cnii,2)
@ 20,30 say str(cfii/cniia,2)
@ 20,37 say str(csii/cniia,2)
@ 21,15 say "III"
@ 21,22 say str(cniii,2)
@ 21,30 say str(cfiii/cniiia,2)
@ 21,37 say str(csiii/cniiia,2)
@ 22,16 say "IV"
@ 22,22 say str(cniv,2)
@ 22,30 say str(cfiv/cniva,2)
@ 22,37 say str(csiv/cniva,2)
@ 24, 0 say "Total Test"
@ 24,22 say str(cntot,2)
@ 24,30 say str(cftot/cntota,2)
@ 24,37 say str(cstot/cntota,2)
release all like cn*
release all like cf*
release all like cs*
return
```

```
*  tssprint.prg     01/31/84  jjb
*  called by draft.prg

*  prints a Test Summary Sheet summarizing data on all the items in the
*  test under development.

*  set up the memory variables
release all except st*
restore from b:workm.mem additive
store "          Printing a Test Summary Sheet" to title
store "ITEMBANK Test Summary Sheet" to topline

*  check to see that a draft test file exists.  if it doesn't, display
*  a warning message.  if it does, remind the user of the draft test form
*  number, administration date, and equator form numbers.
do while more1
     do head2
     if .not. file ("b:test2")
        set color to 14,14
        @  7,10 say "No draft test file has been established."
        @  9,10 say "See your supervisor for further assistance."
        set color to 27,27
        @ 20,10 say "To return to the Draft Stage Menu, press M."
        @ 21,10 say "To quit, press any other key."
        set console off
        wait to answer
        set console on
        if !(answer) = "M"
           return
        else
           do end
        endif
     else
        do testinfo
        set color to 10,10
        @ 17,10 say "This step will print the Test Summary Sheet for " +;
                    "the draft version"
        @ 18,10 say "of the test currently under development."
        set color to 27,27
        @ 20,10 say "To return to the Draft Stage Menu, press M."
        @ 21,10 say "To continue, press any other key."
        set console off
        wait to answer
        set console on
        if !(answer) = "M"
           return
        else
           store f to more1
        endif
     endif
enddo

*  allow the user to select in what order the items will be listed.
do while more2
     do head2
```

```
use b:test2
set color to 10,10
@  7,10 say "Enter the letter that represents the order in " + ;
          "which you want "
@  8,10 say "the items listed.  Select only one."
@ 10,15 say "A = by item ID number;"
@ 12,15 say "B = by book number (normally equivalent to listing by"
@ 13,15 say "    content classification, function and ID number;"
@ 15,15 say "C = by function, content classification, and ID number."
set console off
wait to answer
set console on
store !(answer) to answer
do message2
do case

   case answer = "A"
   index on id to b:test2x

   case answer = "B"
   index on tbook to b:test2x

   case answer = "C"
   index on tfunc + cont + id to b:test2x

   otherwise
   do error
   loop

endcase
use b:test2 index b:test2x
do message1
do message2
set format to print
do head3
set margin to 5
do while .not. eof
   if ctr2 >= 60
      do newpage
   endif
   do print3
   skip
   loop
enddo
eject
do newpage
do testsum
set print on
eject
set print off
delete file b:test2x.ndx
delete file b:temp
set format to screen
store ' ' to answer
store t to more3
```

```
      do while more3
         do head2
         set color to 10,10
         @  7,10 say "What do you want to do next?  Select only one."
         @  9,15 say "P = print another Test Summary Sheet;"
         @ 11,15 say "M = return to the Draft Stage Menu;"
         @ 13,15 say "Q = quit."
         set console off
         wait to answer
         set console on
         store !(answer) to answer
         do case

            case answer = "P"
            store f to more3

            case answer = "M"
            release all except st*
            return

            case answer = "Q"
            do end

            otherwise
            do error
            loop

         endcase
      enddo
      loop
enddo
```

```
*  working.prg      02/01/84  jjb
*  called from itembank.prg

 erase
set color to 12,12
@  1,21 say "======================================"
@  5,21 say "======================================"
set color to 11,11
@  3,21 say "WORKING STAGE MENU -- ITEMBANK SYSTEM"
set color to 10,10
@  5,80
text
Following the client review, the draft test becomes a working test.  Changes

recommended by reviewers are incorporated in the draft examination beginning

with the substitution of alternate items selected by reviewers to replace

rejected items from the draft test.  Data on new items not previously used o

any test form are entered to the database.  Programs in this stage assist in

the assembly and revision of the test.  The stage ends when the test, with t

items in test order, is sent to reviewers for comments and changes.
endtext
set color to 14,14
@ 21,19 say "Programs for this stage are not available."
set color to 27,27
@ 23,19 say "Press any key to return to the Main Menu."
set console off
wait
set console on
return
```